

Project Number:	604102	Project Title:	Human Brain Project
Document Title:	Neuroinformatics Platform v1 - specification document		
Document Filename ⁽¹⁾ :	SP5_D5.8.1_Resubmission_FINAL.docx		
Deliverable Number:	D5.8.1		
Deliverable Type:	Report (Platform Specification)		
Work Package(s):	WP5.1, WP5.2, WP5.3, WP5.4, WP5.5, WP5.6, WP5.7, WP5.8		
Dissemination Level:	PU		
Planned Delivery Date:	M6/31 March 2014		
Actual Delivery Date:	M13/30 October 2014, RESUBMITTED M28/04 Jan 2016		
Authors:	Sean HILL, EPFL (P1); Sten GRILLNER, KI (P31)		
Contributors:	EPFL (P1): Catherine ZWAHLEN CNRS (P7): Andrew DAVISON JÜLICH (P17): Michael DENKER, Sonja GRÜN, Alper YEGENOGLU SKU (P49): Paul TIESINGA UPM (P59): José Maria PEÑA, Pedro LARRAÑAGA UIO (P68): Jan BJAALIE		
Editors:	EPFL (P1): Guy WILLIS, Lauren ORWIN UHEI (P45): Martina SCHMALHOLZ		
Abstract:	<p>This Deliverable sets out the specifications for the HBP's Neuroinformatics Platform (NIP), which serves as a central brain data resource for the Project's other five Platforms. The principle elements of the NIP defined here are:</p> <ul style="list-style-type: none"> • Tools for Brain Atlases (including the Brain Atlas Builder). • Tools for Structural Data Analysis • Tools for Functional Data Analysis • Predictive Neuroinformatics • Brain Atlases (Mouse and Human) <p>The document also describes the overall architecture of the NIP, provides a roadmap for the assembly of the atlases, and defines key performance indicators (KPIs) for monitoring progress.</p>		
Keywords:	NIP, brain, atlases, data, platform, specification		

Table of Contents

Executive Summary	6
1. Introduction.....	7
1.1 The Human Brain Project (HBP)	7
1.2 The Neuroinformatics Platform (NIP): Operational Objectives	7
1.3 NIP: Overall Architecture	8
1.4 NIP Subproject, Work Package and Task Organisation	10
1.4.1 Subproject Leadership	10
1.4.2 Work Package and Task Leadership	10
1.5 NIP Deliverables: HBP Ramp-Up Phase	11
1.6 NIP Interdependencies with other Subprojects	12
1.7 Purpose of this Document	12
1.8 Documentation	12
2. Tools for Brain Atlases (WP5.1)	14
2.1 Tools for Brain Atlases: Data Management Tools.....	14
2.1.1 Tools for Brain Atlases: Use Cases	14
2.1.2 Tools for Brain Atlases: Functional Requirements.....	19
2.1.3 Tools for Brain Atlases: Use Case to Functional Requirement Mapping Table	26
2.1.4 Tools for Brain Atlases: Non-Functional Requirements.....	26
2.1.5 Tools for Brain Atlases: Software	26
2.1.6 Tools for Brain Atlases: Physical Architecture	27
2.1.7 Tools for Brain Atlases: Services required from other Platforms	27
2.1.8 NIP: Services provided to other Platforms.....	27
2.1.9 Tools for Brain Atlases: Prerequisites	27
2.1.10 Tools for Brain Atlases: Necessary Parallel Activities	27
2.1.11 Tools for Brain Atlases: Hardware/Software Functions	28
2.2 Brain Atlas Builder (T5.1.6)	30
2.2.1 Brain Atlas Builder: Overall Goals	30
2.2.2 Brain Atlas Builder: Use Cases	30
2.2.3 Brain Atlas Builder: Functional Requirements	32
2.2.4 Brain Atlas Builder: Non-Functional Requirements	33
2.2.5 Brain Atlas Builder: Software	33
2.2.6 Brain Atlas Builder: Physical Architecture	33
2.2.7 Brain Atlas Builder: Services required from other Platforms.....	33
2.2.8 Brain Atlas Builder: Services provided to other Platforms.....	34
2.2.9 Brain Atlas Builder: Prerequisites	34
2.2.10 Brain Atlas Builder: Necessary Parallel Activities	34
3. Tools for Structural Data Analysis (WP5.2 and WP5.4).....	35
3.1 Structural Tools: EMDigest (T5.2.1)	35
3.1.1 EMDigest: Overall Goals	35
3.1.2 EMDigest: Use Cases	35
3.1.3 EMDigest: Functional Requirements	36
3.1.4 EMDigest: Non-Functional Requirements.....	37
3.1.5 EMDigest: Software	37
3.1.6 EMDigest: Physical Architecture	37
3.1.7 EMDigest: Services required from Other Platforms	37
3.1.8 EMDigest: Prerequisites	38
3.1.9 EMDigest: Necessary Parallel Activities	38
3.2 Structural Tools: EspINA (T5.2.2)	38
3.2.1 EspINA: Overall Goals.....	38

3.2.2	EspINA: Use Cases	38
3.2.3	EspINA: Functional Requirements	44
3.2.4	EspINA: Non-Functional Requirements	49
3.2.5	EspINA: Software	50
3.2.6	EspINA: Physical Architecture.....	50
3.2.7	EspINA: Services required from other Platforms	50
3.2.8	EspINA: Services provided to other Platforms	51
3.2.9	EspINA: Prerequisites	51
3.2.10	EspINA: Necessary Parallel Activities	51
3.3	Structural Tools: 3DSynapsesSA (T5.4.2)	51
3.3.1	3DSynapsesSA: Overall Goals.....	51
3.3.2	3DSynapsesSA: Use Cases	51
3.3.3	3DSynapsesSA: Functional Requirements	52
3.3.4	3DSynapsesSA: Non-Functional Requirements	52
3.3.5	3DSynapsesSA: Software	53
3.3.6	3DSynapsesSA: Physical Architecture.....	53
3.3.7	3DSynapsesSA: Services required from other Platforms.....	53
3.3.8	3DSynapsesSA: Services provided to other Platforms.....	53
3.3.9	3DSynapsesSA: Prerequisites	53
3.3.10	3DSynapsesSA: Necessary Parallel Activities	54
3.3.11	3DSynapsesSA: Hardware/Software Functions	54
3.4	Structural Tools: 3DPyrStructure (T5.4.2).....	55
3.4.1	3DPyrStructure: Overall Goals	55
3.4.2	3DPyrStructure: Use Cases	55
3.4.3	3DPyrStructure: Functional Requirements.....	56
3.4.4	3DPyrStructure: Non-Functional Requirements	57
3.4.5	3DPyrStructure: Software.....	57
3.4.6	3DPyrStructure: Physical Architecture	57
3.4.7	3DPyrStructure: Services required from other Platforms	58
3.4.8	3DPyrStructure: Services provided to other Platforms	58
3.4.9	3DPyrStructure: Prerequisites	58
3.4.10	3DPyrStructure: Necessary Parallel Activities.....	58
3.4.11	3DPyrStructure: Hardware/Software Functions	58
3.5	Structural Tools: 3DSomaMS (T5.4.2).....	61
3.5.1	3DSomaMS: Overall Goals.....	61
3.5.2	3DSomaMS: Use Cases	61
3.5.3	3DSomaMS: Functional Requirements	62
3.5.4	3DSomaMS: Non-Functional Requirements.....	63
3.5.5	3DSomaMS: Software	64
3.5.6	3DSomaMS: Physical Architecture	64
3.5.7	3DSomaMS: Services required from other Platforms.....	64
3.5.8	3DSomaMS: Services provided to other Platforms.....	64
3.5.9	3DSomaMS: Prerequisites	64
3.5.10	3DSomaMS: Necessary Parallel Activities	64
3.5.11	3DSomaMS: Hardware/Software Functions	65
4.	Tools for Functional Analysis (WP5.3)	67
4.1	Functional Data Analysis Tools (FDAT): Overall Goals	67
4.1.1	FDAT: Use Cases.....	67
4.1.2	FDAT: Functional Requirements	74
4.1.3	FDAT: Non-Functional Requirements	76
4.1.4	FDAT: Software	77



4.1.5	FDAT: Physical Architecture.....	78
4.1.6	FDAT: Services required from other components of the NIP	78
4.1.7	FDAT: Services required from other Platforms	78
4.1.8	FDAT: Services provided to other Platforms	79
4.1.9	FDAT: Prerequisites.....	79
4.1.10	FDAT: Necessary Parallel Activities	79
4.1.11	FDAT: Hardware/Software Functions.....	79
5.	Predictive Neuroinformatics (WP5.4)	90
5.1	Predictive Neuroscience: Brain Addressing System	90
5.1.1	Brain Addressing System: Overall Goals.....	90
5.1.2	Brain Addressing System: Use Cases	90
5.1.3	Brain Addressing System: Functional Requirements	91
5.1.4	Brain Addressing System: Non-Functional Requirements	93
5.1.5	Brain Addressing System: Software	93
5.1.6	Brain Addressing System: Physical Architecture	94
5.1.7	Brain Addressing System: Services required from other Platforms.....	94
5.1.8	Brain Addressing System: Services provided to other Platforms.....	94
5.1.9	Brain Addressing System: Prerequisites	94
5.1.10	Brain Addressing System: Necessary Parallel Activities	95
5.1.11	Brain Addressing System: Hardware/Software Functions	95
6.	Brain Atlases (WP5.5).....	99
6.1	The Mouse Brain Atlas (T5.5.1)	99
6.1.1	The Mouse Brain 3D Template Package	99
6.1.2	The Rat Brain 3D Template Package.....	99
6.1.3	Rodent Brain Image Data Collections.....	100
6.1.4	Mouse Brain Atlas: Hardware/Software Functions	102
6.2	The Human Brain Atlas (T5.5.2)	103
6.2.1	BigBrain Atlas for High-Resolution Microscopical Data	103
6.2.2	Montreal Neurological Institute Reference Space (MNI space)	104
6.2.3	Allen Human Brain Atlas (AHBA).....	104
6.2.4	Data from the JuBrain Atlas.....	105
7.	Roadmap for Brain Data Integration and Navigation.....	106
8.	Key Performance Indicators (KPIs).....	107
9.	Glossary	108

Table of Figures

Figure 1: Overview of the functions provided by the Neuroinformatics Platform Architecture.	8
Figure 2: Subproject Leadership	10
Figure 3: Work Package and Task Leadership.....	11
Figure 4: NIP Deliverables	12
Figure 5: NIP Interdependencies with other Subprojects.....	12
Figure 6: Example of spatial and semantic metadata annotations necessary for registering a neuron morphology artefact.	16
Figure 7: KnowledgeGraph, based on HBP-CORE metadata specification and data model.....	20
Figure 8: Tools for Brain Atlases: Use Case to Functional Requirement mapping	26
Figure 9: Brain Atlas Builder	33
Figure 10: Roadmap for brain data integration and navigation	106
Figure 11: Key Performance Indicators (KPIs)	107



Executive Summary

This report provides a detailed specification of the Neuroinformatics Platform (NIP) Version 1, the capabilities of the Platform and the first draft brain atlases. Additionally, the report provides indicators of progress.

The NIP serves as a central resource for the Collaboratory, the Brain Simulation Platform (BSP) and other Platforms to register, search and access multi-level brain data, models and literature.

The NIP is built around a core database that tracks high-level metadata and provenance information for all data elements throughout the Project. The high-level metadata are defined in ontologies that are made public and are maintained and curated in the KnowledgeSpace. The KnowledgeSpace also serves as a repository of “living review articles” that summarise and link to currently available data, models and literature.

The atlas builder provides an important path to registering and curating data, models and literature into standard brain atlases. Text mining provides an important tool for identifying literature related to specific concepts (i.e., brain areas, connectivity, neuron types, etc.) and populating the brain atlases with the results. Analysis and prediction tools draw upon data from the brain atlases and register their output back to the relevant atlases.

Overall, the NIP provides core data registration, atlas navigation, search, integration, analysis and prediction services to the Collaboratory and BSP.

The NIP relies heavily on the Collaboratory to integrate many of its components. For this reason, we refer the reader to the specification of the Collaboratory in the Brain Simulation Platform specifications document.

1. Introduction

1.1 The Human Brain Project (HBP)

The Human Brain Project (HBP) is a major international scientific research project, involving over 100 academic and corporate entities in more than 20 countries. Funded by the European Commission (EC), the ten-year, EUR 1 billion Project was launched in 2013 with the goal "to build a completely new ICT infrastructure for neuroscience, and for brain-related research in medicine and computing, catalysing a global collaborative effort to understand the human brain and its diseases and ultimately to emulate its computational capabilities."

The fields of neuroscience, medicine and information technology each have important roles to play in addressing this challenge, but the knowledge and data that each is generating have been very fragmented. The HBP is driving integration of these different contributions.

During the Ramp-Up Phase, the HBP will collect strategic data, develop theoretical frameworks, and perform technical work necessary for the development of six Information and Communication Technology (ICT) Platforms during the Operational Phase. The ICT Platforms, offering services to neuroscientists, clinical researchers and technology developers, comprise Neuroinformatics (a data repository, including brain atlases and analysing tools); Brain Simulation (building ICT models and multi-scale simulations of brains and brain components); Medical Informatics (bringing together information on brain diseases); Neuromorphic Computing (ICT that mimics the functioning of the brain); and Neurorobotics (allowing testing of brain models and simulations in virtual environments). A High Performance Computing Platform will support these Platforms.

1.2 The Neuroinformatics Platform (NIP): Operational Objectives

One of the HBP's most important objectives is to make it easier for neuroscientists to organise and access the massive volumes of heterogeneous data, knowledge and tools produced by the international neuroscience community - a goal it shares with the INCF and with other on-going projects, in particular the Allen Institute's Brain Atlas projects (www.brain-map.org). The Neuroinformatics Platform will contribute to these efforts, offering new tools for the construction of multi-level brain atlases (WP5.1) and for the analysis and interpretation of large volumes of structural (WP5.2) and functional (WP5.3) data. The HBP will use these tools to develop detailed multi-level atlases of the rodent and human brains (WP5.5), bringing together data from the literature, and from on-going research. It will provide a single source of annotated, high quality data for the HBP modelling effort and for the international neuroscience community. Another key feature of the Platform will be support for Predictive Neuroinformatics (WP5.4): the mining of large volumes of data, and the analysis of activity data to identify patterns and relationships between data from different levels of biological organisation. Predictive neuroinformatics will make it possible to predict parameters where experimental data are not yet available and to test and calibrate model implementations. Systematic application of this strategy has the potential to drastically increase the amount of information that can be extracted from experimental data, rapidly filling gaps in our current knowledge and accelerating the generation of data required for brain modelling.

The objective for the Ramp-Up Phase is to launch the first functional version of the Neuroinformatics Platform and populate it with data, models and ontologies for ion

channels, cell types, synapse types and microcircuits. The Platform will include a Neuroinformatics web portal for accessing the Platform - including atlases, ontologies and tools. The data and models will be placed in the Data space and annotated with ontologies from the Brainpedia. Initial tools and workflows for analysis of electron microscopy data and local field potential data will be deployed. A core service for tracking provenance of data and models throughout the workflows will ensure reproducibility and appropriate attribution. Our initial Predictive Neuroinformatics efforts will target the prediction of fine-scale and long-range connectivity from DTI imaging data.

1.3 NIP: Overall Architecture

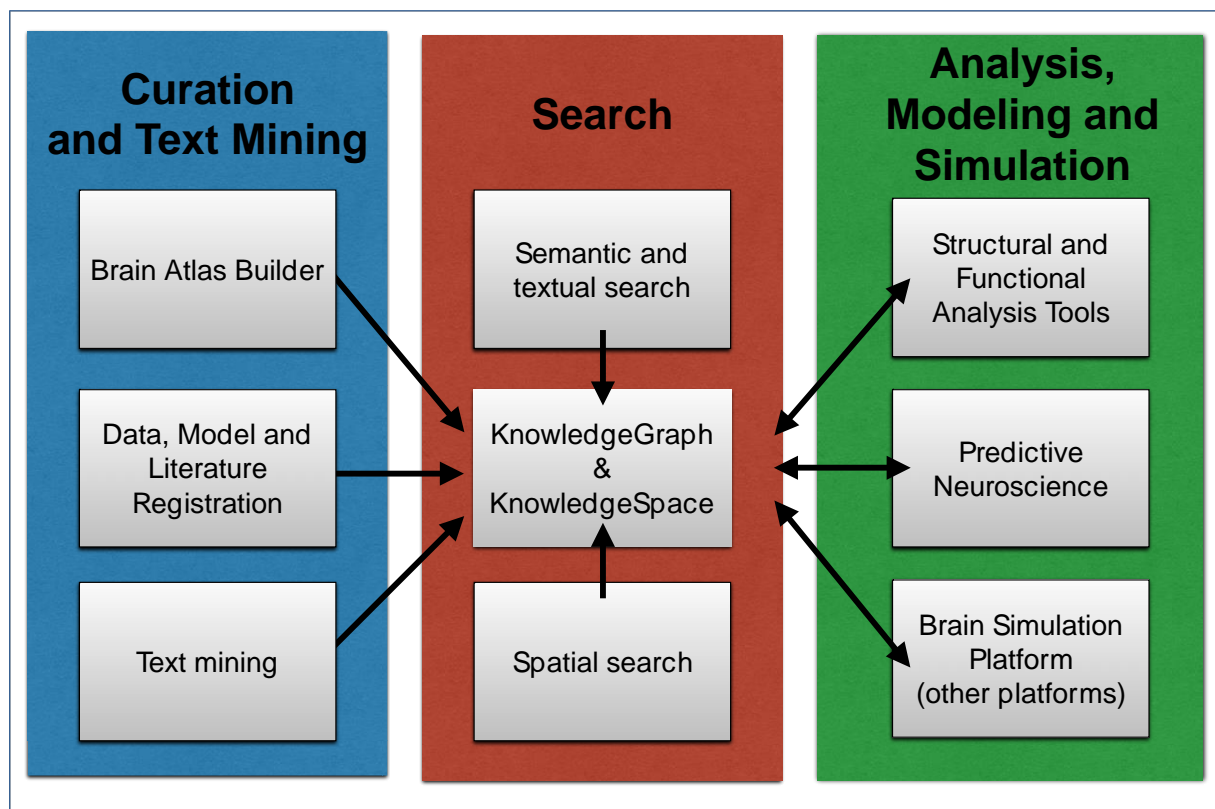


Figure 1: Overview of the functions provided by the Neuroinformatics Platform Architecture.

Data are curated and registered into the KnowledgeGraph and KnowledgeSpace or LabSpace via the Brain Atlas Builder, manual data, model and literature registration workflows, and automatic text mining. Analysis, modelling and simulation tools use an array of textual, semantic and spatial search services to find data to work with. These same tools register their results in the KnowledgeGraph and KnowledgeSpace, as appropriate.

The Neuroinformatics Platform is intended to make a large body of neuroscience data easily discoverable and accessible to the scientific community. The Platform provides tools to compare and analyse data sets, allowing scientists to derive new knowledge about the brain.

A unified description of neuroscience data is crucial to achieving meaningful data integration, and enabling powerful searching. For this purpose, a common data model captures key aspects of experimental data sets, calculated models, and information extracted from the literature. The model is built based on ontologies, which provide a straightforward mechanism to integrate new information and allow for the model to evolve over time. Moreover, the definition and implementation of data standards is a key



component to scale the data integration. Finally, the Platform provides programmatic and graphic interfaces to register data in this common framework.

The NIP serves as a central resource for all HBP Platforms to register, curate, track, search and analyse every element of data of interest - be it from experiment, software or hardware simulation, or literature. All data are registered in the NIP with a set of minimal metadata, including provenance information about the specimen (biological or *in silico*), contributors, methods, brain location and data type.

A key part of the data registration process is integrating the data within existing brain atlases, either through providing precise spatial coordinates or a standardised name of a brain region. This involves ensuring that the data are anchored within a standard brain-based coordinate system, and ensuring that brain region names conform to standard ontologies for well-defined brain atlases.

Text mining is an important service provided by the Neuroinformatics Platform, as it provides the ability to parse large volumes of literature and identify types of content contained within it. Text mining services are used to populate brain atlases with references to relevant literature related to a given brain structure or location.

Once the data are registered and integrated within the Platform, they can be queried for analysis, modelling and prediction. The semantic definition of the metadata is key to enabling versatile search and identifying data that can be integrated and analysed together. Data curation involves ensuring accurate annotation of all high-level metadata, and verifying the quality of the data.

Three areas of analysis and modelling are also represented in the Neuroinformatics Platform. These include analysis of structural data (including neuron morphology, subcellular properties and other brain and cellular level structural properties); analysis of functional data (including signal analysis, spike sorting and intracellular recordings); and predictive neuroscience (including the prediction of connectivity between neurons throughout the brain). Each of these represents key areas of neuroinformatics and is represented in the initial version of the Platform only by a few examples. Ultimately, the Platform should facilitate integration of many such tools and workflows.

Predictive neuroscience represents a fundamental goal of the entire data curation, integration and analysis process. Predictive neuroscience involves using sparse samples of data to make predictions and fill in missing data. There are many structural and functional scales to brain organisation. The predictive neuroscience in the Neuroinformatics Platform will be used to make new predicted brain atlases for structural brain data. Predictions of functional data remain the purview of the Brain Simulation Platform.

The Platform catalyses collaboration through the use of common tools and data curation interfaces. Furthermore, detailed brain atlases provide powerful spatial navigation through large amount of information and facilitate data discovery.

To enable these goals, a number of different components are being built and will be made available:

- **Tools for Brain Atlases (Work Packages 5.1 & 5.6 - Chapter 2)**
 - Shared DataSpace: a global data federation allowing the registration of data sets from distributed data repositories.
 - KnowledgeGraph: representation of a common data model that encapsulates high-level metadata for data stored in the DataSpace and elsewhere.
 - Ontologies and data standards: develop and curate common ontologies and data standard for project wide use.
 - Data mining infrastructure: find and analyse neuroscience data.
 - Brain Atlas Builder: develop tools to organise and anchor data to standard anatomical ontologies and 3D reference spaces.
- **Tools for Structural Data Analysis (Work Package 5.2 - Chapter 3)**
 - Analysis of experimental data obtained by EM and light microscopy (EMDigest)
 - Segmentation and annotation of brain tissue microscopy stacks (EspINA).
 - Analysis of synapse distribution (3DSynapsesSA).
 - Reconstruction of neuron morphology of neurons (3DPyrStructure).
 - Reconstruction of soma morphology (3DSomaMS).
- **Tools for Functional Data Analysis (Work Package 5.3 - Chapter 4)**
 - Analysis of multi-scale brain dynamics data (Functional Data Analysis Toolbox FDAT)
- **Predictive Neuroinformatics (Work Package 5.4 - Chapter 5)**
 - Connection matrices between neuron populations (Brain Addressing System).
- **Brain Atlases (Work Package 5.5 - Chapter 6)**
 - Mouse Brain Atlas.
 - Human Brain Atlas.

The Neuroinformatics Platform provides an informatics infrastructure and analysis tools. The Platform must serve both power users and casual users, because the goals of the Human Brain Project can only be met by active collaboration between biologists, scientific modellers and computer scientists.

1.4 NIP Subproject, Work Package and Task Organisation

1.4.1 Subproject Leadership



#	Subproject Name	Leader	Partner	State
SP5	Neuroinformatics Platform	Sten GRILLNER	KI (P31)	
SP5	Neuroinformatics Platform	Sean HILL	EPFL (P1)	

Figure 2: Subproject Leadership

1.4.2 Work Package and Task Leadership

#	Work Package/Task Name	Leader/Grant Recipient(s)	Partner	State
---	------------------------	---------------------------	---------	-------

WP5.1	Tools for Brain Atlases	Sean Hill	EPFL (P1)	
T5.1.1	Shared Data Space	Sten Grillner	KI (P31)	
T5.1.2	Data Mining	Sean Hill	EPFL (P1)	
T5.1.3	Ontologies	Sten Grillner Maryann Martone	KI (P31) UCAL (P41)	 
T5.1.4	Data Standards	Sten Grillner	KI (P31)	
T5.1.5	KnowledgeSpace (formerly Brainpedia)	Sten Grillner	KI (P31)	
T5.1.6	3D Brain Atlas Builder	Sean Hill Jan Bjaalie Karl Zilles Katrin Amunts	EPFL (P1) UIO (P68) JÜLICH (P17) JÜLICH (P17)	   
WP5.2	Tools for Structural Data Analysis	José M. Peña	UPM (P59)	
T5.2.1	EMDigest	José M. Peña	UPM (P59)	
T5.2.2	Segment	Pascal Fua	EPFL (P1)	
WP5.3	Tools for Functional Data Analysis	Sonja Grün	JÜLICH (P17)	
T5.3.1	Population analysis	Sonja Grün	JÜLICH (P17)	
T5.3.2	Cell analysis	Andrew Davison	CNRS (P7)	
WP5.4	Predictive Neuroinformatics	Paul Tiesinga	SKU (P49)	
T5.4.1	Neuronal addressing system	Paul Tiesinga	SKU (P49)	
T5.4.2	Neuronal structural design and predictions	Pedro Larranaga	UPM (P59)	
WP5.5	Brain Atlases	Jan BJAALIE	UIO (P68)	
T5.5.1	The Mouse Brain Atlas	Jan Bjaalie	UIO (P68)	
T5.5.2	The Human Brain Atlas	Katrin Amunts	JÜLICH (P17)	
WP5.6	Neuroinformatics Platform: Integration and Operations	Sean HILL	EPFL (P1)	
T5.6.1	Integration, website construction, maintenance and administration	Sean Hill	EPFL (P1)	
WP5.7	Neuroinformatics Platform: User Support and Community Building	Sten Grillner	KI (P31)	
T5.7.1	The Neuroinformatics service centre for user training and user support	Sten Grillner	KI (P31)	
T5.7.2	International alignment	Sten Grillner	KI (P31)	
WP5.8	Neuroinformatics Platform: Scientific Coordination	Sean HILL	EPFL (P1)	
T5.8.1	Scientific coordination and support	Sean Hill	EPFL (P1)	

Figure 3: Work Package and Task Leadership

1.5 NIP Deliverables: HBP Ramp-Up Phase

Due	D No.	Deliverable	Leader	State
-----	-------	-------------	--------	-------






Month 6	D5.8.1	Neuroinformatics Platform v1 - specification document (report)	Sean HILL	
Month 12	D5.8.2	Neuroinformatics Platform v1 - set-up document (report)	Sean HILL	
Month 18	D5.8.3	Neuroinformatics Platform v1 - preliminary release for internal Consortium use (prototype)	Sean HILL	
Month 30	D5.8.4	Neuroinformatics Platform v1 (prototype?)	Sean HILL	
Month 30	D5.8.5	Neuroinformatics Platform v1 - documentation (report)	Sean HILL	

Figure 4: NIP Deliverables

1.6 NIP Interdependencies with other Subprojects

SP5 and the Neuroinformatics Platform interact closely with the following other HBP SPs:

SP	SP Name	Provides NIP with	Is provided by NIP with
1	Strategic Mouse Brain Data	Mouse brain data	
2	Strategic Human Brain Data	Human brain data	
3	Cognitive Architectures	Functional data	
4	Theoretical Neuroscience	Predictive models	Data Models Curated literature
6	Brain Simulation Platform	Models Simulation output Analysis results	Data for models
7	High Performance Computing Platform	Storage Analysis platforms	
8	Medical Informatics Platform	Aggregate phenotype descriptions Disease definitions	Ontology services KnowledgeSpace
9	Neuromorphic Computing Platform		Models Data Curated literature
10	Neurorobotics Platform		Models Data Curated literature

Figure 5: NIP Interdependencies with other Subprojects

1.7 Purpose of this Document

This report will provide a detailed specification of the Neuroinformatics Platform v1, the tools incorporated in the Platform, the two Brain Atlases, and planned Predictive Neuroscience informatics methods. Additionally, the report will specify indicators of progress and target values for the indicators.

1.8 Documentation

The documentation for the Neuroinformatics Platform is available on the website <https://nip.humanbrainproject.eu/documentation.html>, and is kept up-to-date to

reflect new developments, and updates of data types and data standards.

The mission of the Neuroinformatics Platform is to provide access to integrated and curated data. A minimal set of metadata, HBP-Core (<https://nip.humanbrainproject.eu/documentation/hbp-core.html>), is specified when data are ingested in the Platform's KnowledgeGraph. The ontologies used for curation are published in a GitHub repository: <https://github.com/NeuroscienceKnowledgeSpace/methodsOntology>. As the ontologies are updated, they automatically become available, and new terms can be used to specify metadata. The system in place is open and flexible, and facilitates contributions from all members of the Project.

Finally, the schema for data registration, as well as the schemas for search results, are published in GitHub: <https://github.com/HumanBrainProject/hbp-data-schemas>.



2. Tools for Brain Atlases (WP5.1)

The HBP will create a general-purpose open-source software framework, allowing researchers to build and navigate multi-level atlases of the brain of any species. These tools will allow researchers to upload and access multi-level information about any part of the brain. The information will provide a shared data space (T5.1.1), data mining tools (T5.1.2, T5.1.3), data standards (T5.1.4) and a generic “Atlas Builder” (T5.1.6) making it possible to build, manage and query such atlases. In addition to this work, the project will also create and manage an HBP “Knowledge Space” - a community-driven Wiki that provides an encyclopaedic view of the latest data, models and literature for all levels of brain organisation (T5.1.5).

2.1 Tools for Brain Atlases: Data Management Tools

This section covers all the data management tools in the Tools for Brain Atlases, except for the Brain Atlas Builder, which is the subject of the following section. This section deals with a range of activities where there are many interdependencies: data federation, data registration, data annotation, data curation, and text mining and search functions.

2.1.1 Tools for Brain Atlases: Use Cases

The Use Cases below describe success scenarios for small groups of scientific users. The scenarios describe high-level interaction with the Brain Simulation Platform and are dependent on functionality provided by many of the Brain Simulation Platform’s components and its dependencies.

2.1.1.1 Federating neuroscience data repositories

Use Case ID:

- SP5NIP-UC-001

Primary Actors:

- Scientific research group Ajax

Success Scenario:

- Scientific group Ajax is producing data it would like to share with the HBP Consortium or other collaborators.
- The technical contact from Ajax works with the DataSpace team to install and configure DataSpace software.
- The Ajax team federates their data with the global DataSpace federation (managed by INCF).
- The Ajax team can set read/write privileges to allow global access to their data repository through the INCF login.
- Scientific user Bill can now download or uses Portal tasks to work with the morphologies.

2.1.1.2 Registering data sets

Use Case ID:



- SP5NIP-UC-002

Primary Actors:

- Scientific research group

Success Scenario:

- Scientific group Ajax would like to explicitly register an existing data set to make it available to the DataSpace federation.
- The Ajax team executes a register command on the data set, which adds the essential metadata to the global metadata catalogue.
- Other members of the data federation can now see and access the data set.

2.1.1.3 Annotating data from an XML template

Use Case ID:

- SP5NIP-UC-003

Primary Actors:

- Scientific research group

Success Scenario:

- Scientific group Ajax would like to annotate a data set with free text and metadata - providing the context of data acquisition and sufficient metadata to characterise measurement parameters.
- The Ajax team creates an XML template file containing the metadata in a standard form.
- A script is executed which parses the XML metadata file and populates a wiki page with a template page describing the data, using the metadata tags.
- The Ajax team can further annotate the data with text in a wiki environment.

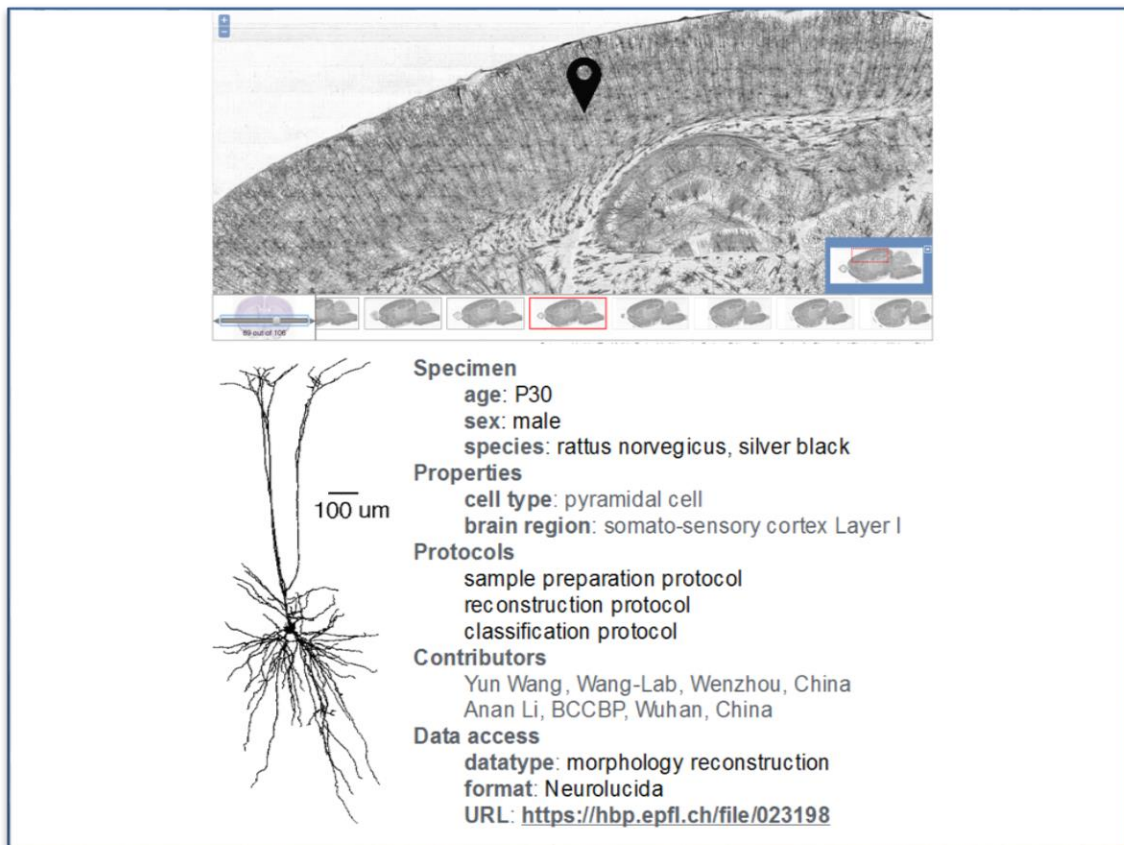


Figure 6: Example of spatial and semantic metadata annotations necessary for registering a neuron morphology artefact.

2.1.1.4 Registering and annotating data through a web interface

Use Case ID:

- SP5NIP-UC-004

Primary Actors:

- Scientific research group

Success Scenario:

- Scientific group Ajax would like to annotate a data set with free text and metadata - providing the context of data acquisition and sufficient metadata to characterise measurement parameters.
- The Ajax team selects a location in an XML template file containing the metadata in a standard form.
- A script is executed which parses the XML metadata file and populates a wiki page with a template page describing the data, using the metadata tags.
- The Ajax team can further annotate the data with text in a wiki environment.



2.1.1.5 Data curation workflow

Use Case ID:

- SP5NIP-UC-005

Primary Actors:

- Data curation team

Success Scenario:

- Data curation team Omega establishes a standardised curation workflow for data from scientific group Ajax.
- The Omega team defines a standard XML metadata template file for scientific group Ajax.
- Ajax submits initial annotated data to the wiki using an annotation workflow.
- Omega team reviews data for completeness and quality.
- Omega team edits KnowledgeSpace entries to provide rich annotation around key data sets and ensures all tags conform to standard ontological terminology, data types, previews, etc.

2.1.1.6 Data type definition registration

Use Case ID:

- SP5NIP-UC-006

Primary Actors:

- Data curation team

Success Scenario:

- Data curation team Omega curates data with a previously unused data type.
- Omega creates a data type definition in the Knowledge Graph.
- The data type is defined within a data type ontology (relating the new data type to other key types - image, waveform, event, video, sound, etc.).

2.1.1.7 Text mining services

Use Case ID:

- SP5NIP-UC-007

Primary Actors:

- Data mining team

Success Scenario:

- Text mining team Alpha extracts facts, assertions, parameters, etc. from published literature.
- The text mining service uses common ontological terms to populate the KnowledgeSpace with the mined data.
- The mined data are annotated with the text mining technique, version and data source.



2.1.1.8 Unified search service

Use Case ID:

- SP5NIP-UC-008

Primary Actors:

- A scientific user

Success Scenario:

- A scientific user wishes to search for data, models or literature.
- The user provides free text or specific semantic tags to define the search.
- The unified search service returns a collection of objects consistent with the search query.

2.1.1.9 Finding data to construct and validate a model of a neuron type

Use Case ID:

- SP5NIP-UC-009

Primary Actors:

- A scientific user

Success Scenario:

- Scientific user Abigail would like to build a model of a neuron type brain model of a cortical pyramidal neuron.
- She searches for all morphological data for cortical pyramidal neurons.
- She searches for all electrophysiological data for cortical pyramidal neurons.
- She searches for all gene expression data for cortical pyramidal neurons.
- She selects the data that has been curated and identified as ready for model building.
- With the search results she now can begin her workflow in the Brain Simulation Platform.
- She uses data that has been curated and identified as ready for validation use to validate the model.

2.1.1.10 Finding data to construct and validate a model of a cortical microcircuit

Use Case ID:

- SP5NIP-UC-009

Primary Actors:

- A scientific user

Success Scenario:

- Scientific user Abigail would like to build a model of a cortical microcircuit.
- She searches for all cell types in a cortical region.
- She searches for cell density data for each cell type, either raw measurement or prediction.



- She searches for models for each cell type.
- She searches for all synapse properties (neurotransmitter type, release probability, short-term plasticity class, etc.) for each connection pathway, measured or predicted.
- She searches for connection probabilities either measured or predicted between all cell types.
- She selects the data that has been curated and identified as ready for model building.
- With the search results she now can begin her workflow in the Brain Simulation Platform.
- She uses data that has been curated and identified as ready for validation use to validate the model.

2.1.1.11 Finding data to construct and validate a model of a whole mouse brain

Use Case ID:

- SP5NIP-UC-009

Primary Actors:

- A scientific user

Success Scenario:

- Scientific user Csaba would like to build a model of a whole mouse brain.
- He searches for cell densities for each brain region.
- He searches for cell-type gene expression markers for each brain region, from literature and gene-expression databases.
- He constructs a predictive model for cell-type densities for each brain region.
- He searches for whole brain DTI data for macro-scale connectivity.
- He searches for whole brain tracer injection data for fine-scale connectivity.
- He constructs a predictive model for whole brain neuron connectivity.
- He selects the data that have been curated and identified as ready for model building.
- He searches for literature-mined properties for each brain area to further parameterise or validate the model.
- With the search results he now can begin his workflow in the Brain Simulation Platform.

2.1.2 Tools for Brain Atlases: Functional Requirements

2.1.2.1 HBP-CORE: Minimal metadata specifications

Functional Requirement ID:

- SP5-FR-001

Features:

Metadata provide essential information about the neuroscientific data shared in DataSpace in order to ensure traceability of any data artefact. The metadata capture high-level

description of experimental procedures, essential details about biological samples and experimental results. Metadata also register scientists involved in data production. Structured metadata facilitate the integration and retrieval of data by defining a common language across many laboratories and experiment types. A large part of the metadata will be specified using ontologies and dictionaries. The amount of mandatory metadata will be kept to a minimum, applicable to all data sets. A standard metadata data model will be defined and will serve as the minimum specification (HBP-CORE, may be referred to as HBPMIN in other specifications) required for all data, models and literature to be accessed via the NIP.

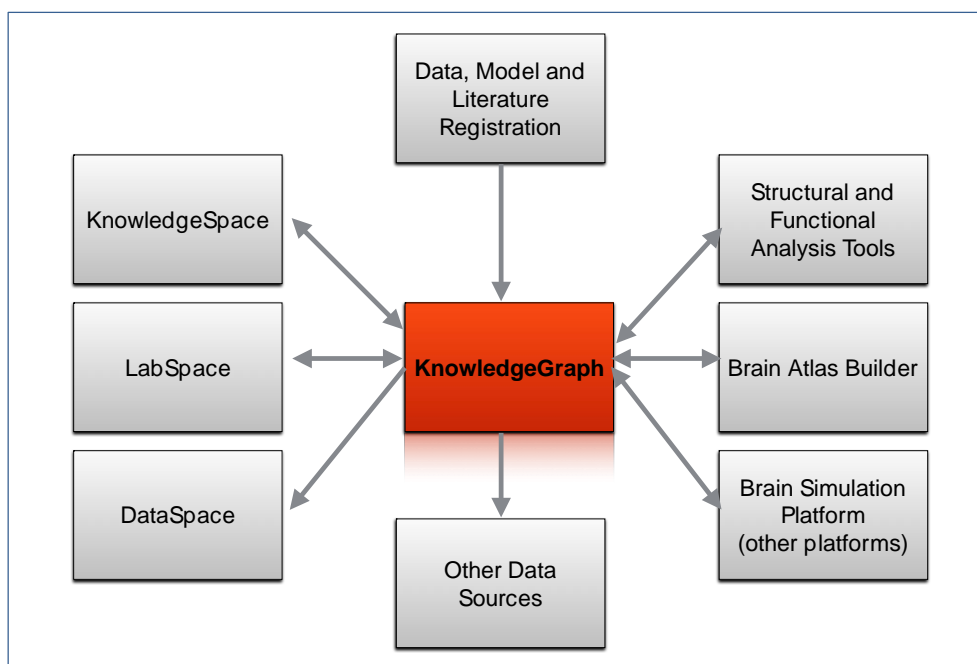


Figure 7: KnowledgeGraph, based on HBP-CORE metadata specification and data model

This serves as the common data store and provenance tracker for all types of data, models, and literature produced and consumed by other services, tools and platforms in HBP.

2.1.2.2 Data standards and data model

Functional Requirement ID:

- SP5-FR-002

Rationale:

Data standards are required to automate the integration processes. The data model represents a unified view of experimental and analysis data and is a prerequisite for developing automated integration processes, as well as for building powerful search engines.

Features:

- Description of data structure/format for all input data types used in the minimum viable product (MVP).
- Data standard easily adopted by scientists.
- Description of a minimal set of experimental metadata to describe experimental materials and methods.
- Define vocabulary and concepts that are part of the data model.



- The data model must be evolvable to include new data types and new requirements.
- Data model must trace the provenance of all data artefacts.

Dependencies:

- Requires defined use cases to identify the different data types and data sources for the MVP.
- Validation of assumptions with scientists uploading the data.

Deliverables:

- HBP-CORE minimal ontology to describe experimental data and data derived from analysis run on the Platform.
- Semantic data model.
- List and sources of necessary ontologies.

2.1.2.3 DataSpace - Data federation of repositories**Functional Requirement ID:**

- SP5NIP-FR-003

Rational:

A means of federating data repositories throughout HBP is required to make data accessible. This data federation constructs a metadata catalogue of all data registered in each repository and provides a means of directly accessing the data through a URI.

Features:

- A metadata catalogue for distributed data.
- An API for registering metadata about the raw data location.
- An authentication/authorisation mechanism.
- A REST-based API to access data.

2.1.2.4 KnowledgeGraph - Database built on HBP-CORE data model and ontologies**Functional Requirement ID:**

- SP5NIP-FR-003

Rationale:

A core database is required to store all data, models and literature and their provenance that will be organised into brain atlases. This database is necessary to implement any search functionality. The database is built on the HBP-CORE data model and thus employs to ontologies to describe and resolve relationships between entries.

Features:

- A graph or relational database designed to store many HBP-CORE described entities.
- Flexible search, semantic, spatial, elastic.
- Rest API for adding, removing, searching entries.

2.1.2.5 KnowledgeSpace - Semantic wiki for community ontology maintenance, data curation and living review articles

Functional Requirement ID:

- SP5NIP-FR-004

Rationale:

The KnowledgeSpace, a public wiki for curation and maintenance of ontologies, data curation and living review articles, will serve as a publication platform. The KnowledgeSpace will be built off of existing infrastructure from the Neuroscience Information Framework (neuinfo.org and neurolex.org).

Features:

- Common neuroscience ontologies and vocabularies.
- Integrated content from multiple organisations including NIF, INCF, BAMS and HBP.
- Semantic queries via SPARQL.
- Public, community-based platform.

2.1.2.6 LabSpace - Semantic wiki for organising and curating summaries of individual laboratory data sets

Functional Requirement ID:

- SP5NIP-FR-005

Rationale:

The LabSpace is a private version of the KnowledgeSpace designed for private registration and curation of individual laboratory data sets. Will support data registration and curation workflows. Once data are curated and ready to be published, they can be replicated to the KnowledgeSpace as a form of publication.

Features:

- Private curation space.
- Web-based registration workflow.
- REST-based API for scripted data registration.

2.1.2.7 Data sharing interface

Functional Requirement ID:

- SP5NIP-FR-006

Rationale:

Using the Collaboratory, scientists can organise and share their experimental, simulation and published results. The process requires specifying metadata associated with the experiments, models and literature. Metadata are ontology driven; however, the user has the possibility to add additional non-structured information.

Features:

- Interface to specify experimental materials and methods, i.e. experimental metadata.
- Upload result files.



- Upload raw files.
- Files are stored in a repository accessible by the integration platform.
- Register uploaded files.

Dependencies:

- HBP-CORE minimal ontology for experimental data.
- Defined data standards for experimental, simulation and literature files.

Deliverables:

- Web interface to specify metadata and upload files.
- Service to store the uploaded data in a file system.
- File registration service.

2.1.2.8 Automated data monitoring pipeline**Functional Requirement ID:**

- SP5NIP-FR-007

Rationale:

For specific data sources, an automated data retrieval process can be developed. The process runs periodically to upload new available data or new version of already existing data.

Features:

- Run periodically.
- Identify if there are new files to upload.
- Upload files and store or cache them on appropriate repository.
- Register newly uploaded files.
- Parameters to specify what to check and where.

Dependencies:

- Infrastructure to store the retrieved data.
- File registration service.

Deliverables:

- Service to upload resources.
- Infrastructure to run the service periodically.

2.1.2.9 Data Integration module**Functional Requirement ID:**

- SP5NIP-FR-008

Rationale:

The integration module is a collection of processes running to consume multiple flows of data, i.e. manually curated or automatically uploaded files. The module is responsible for building any data representation needed by other processes.

**Features:**

- Notifications to alert the integration process when data are available.
- Command line integration to “manually” upload data sets, such as available morphologies.
- Re-usable component to parse data.
- Transform and store data in database according to data model.
- Link data to existing ontologies.
- Old versions of data sets are made obsolete.
- Graph representation of integrated data, i.e. Knowledge Graph.

Dependencies:

- Identified data sources.
- Data model specifications.
- Data standard definitions.
- Defined technologies and application architecture already defined.

Deliverables:

- Notification service to dispatch integration process.
- Service to integrate data from known data types.
- Service to build and update a Knowledge Graph.

2.1.2.10 Data access API**Functional Requirement ID:**

- SP5NIP-FR-009

Rationale:

Once data are stored, they are made available for external use through an API.

Features:

- Build dynamic queries.
- Index data for search.
- “Simple” search, i.e. Google-like search.
- Advanced search.
- Retrieve data in different format

Dependencies:

- A populated database.
- Definition of queries needed by users to specify the entry point of API.
- Identify search use cases.

Deliverables:

- REST API definition.
- Advance search definition.



- Service implementing the API.
- Indexing service.
- Search service.

2.1.2.11 Data analysis and data mining module

Functional Requirement ID:

- SP5NIP-FR-010

Rationale:

The module proposes analysis tools for data sets in the Platform. Results from an analysis may be integrated to enrich the original data.

Features:

- Prepare data in appropriate format for an analysis.
- Define analysis execution plan.
- List of available tools.
- Execute analysis.
- Notify integration service when results are ready.

Dependencies:

- HBP-CORE for analytical data, an extension of HBP-CORE for experimental data.
- API to query and retrieve data stored in the Platform.
- Definition of process to integrate analysis results back in the database.

Deliverables:

- Analytical tools.
- Service to register available tools.
- Service to orchestrate running analytical tools and manage the results.
- Analysis workflow ready to be integrated in Collaboratory using Collaboratory Task object.

2.1.2.12 Text mining module

Functional Requirement ID:

- SP5NIP-FR-011

Rationale:

The module proposes tools for mining features for large bodies of texts, including neuroscience literature.

Features:

- Analyse abstracts, full-text, PDFs.
- Define ontological terms and synonyms to mine.
- Execute analysis.
- Notify integration service when results are ready.

Dependencies:

- HBP-CORE for literature, an extension of HBP-CORE for text mined or curated data.
- API to query and retrieve data stored in the Platform.
- Definition of process to integrate text mining results back in the database.

Deliverables:

- Natural language parsing tools.
- Search service to find and map documents that contain content related to specific ontological concepts.

2.1.3 Tools for Brain Atlases: Use Case to Functional Requirement Mapping Table

SP5NIP-FR-XXX	001	002	003	004	005	006	007	008	009	010	011	012
SP5NIP-UC-001			X	X								
SP5NIP-UC-002	X	X		X								
SP5NIP-UC-003	X	X		X								
SP5NIP-UC-004	X		X	X		X						
SP5NIP-UC-005		X		X		X						
SP5NIP-UC-006				X								
SP5NIP-UC-007	X	X		X								X
SP5NIP-UC-008	X	X		X				X	X			
SP5NIP-UC-009	X	X	X	X	X	X	X	X	X	X	X	X
SP5NIP-UC-010	X	X	X	X	X	X	X	X	X	X	X	X
SP5NIP-UC-011	X	X	X	X	X	X	X	X	X	X	X	X

Figure 8: Tools for Brain Atlases: Use Case to Functional Requirement mapping

2.1.4 Tools for Brain Atlases: Non-Functional Requirements

The NIP depends critically on the data, models and literature to populate it. Access to these will be essential. In addition, defining an open data use policy will be important for successful collaboration on the Platform.

2.1.5 Tools for Brain Atlases: Software

The NIP will be deployed as a set of Web Services.



2.1.6 Tools for Brain Atlases: Physical Architecture

The NIP will be implemented on a distributed set of services deployed on a diverse and dynamic set of hardware architectures spanning cloud, compute cluster and distributed data repositories.

2.1.7 Tools for Brain Atlases: Services required from other Platforms

Collaboratory

2.1.8 NIP: Services provided to other Platforms

- Data federation
- Data registration
- Data annotation
- Data curation
- Data standard definitions
- Data mining services
- Ontology lookup
- Ontology editing
- Ontology curation
- Atlas navigation
- Atlas data anchoring
- Atlas data search
- Atlas curation
- Data analysis services
- Predictive neuroscience services
- Semantic wiki services
- Manual curation workflows
- Wiki documentation of data, model and literature samples

2.1.9 Tools for Brain Atlases: Prerequisites

Data analysis will rely on the Task and Provenance services provided by the Collaboratory and BSP.

2.1.10 Tools for Brain Atlases: Necessary Parallel Activities

Data production from SPs 1, 2, 3 and 4 will be necessary to populate the NIP.

2.1.11 Tools for Brain Atlases: Hardware/Software Functions

Task No:	5.1.4	Partner:	EPFL
Function No:	5.1.4.1	Leader:	Sean Hill
Function Name:	Data standards and data model		
Test Case:	Complete specifications allowing a developer to build an interface for uploading selected data types and their associated metadata.		
Planned Start Date:	May 2014	Planned Completion Date:	April 2015
Requires Functions:	Not applicable		

Task No:	5.6.1	Partner:	EPFL
Function No:	5.6.1.1	Leader:	Sean Hill
Function Name:	Data integration module		
Test Case:	Data are successfully stored in a database with links to the appropriate ontologies and links to the original raw data.		
Test Case:	Data from a specific lab can successfully be retrieved via a database-querying tool.		
Planned Start Date:	June 2014	Planned Completion Date:	December 2014
Requires Functions:	5.1.4.1, T5.1.1 (shared data space)		

Task No:	5.6.1	Partner:	EPFL
Function No:	5.6.1.2	Leader:	Sean Hill
Function Name:	Building a Knowledge Graph		
Test Case:	A user can retrieve all linked information relevant to their query.		
Planned Start Date:	August 2014	Planned Completion Date:	December 2014
Requires Functions:	5.6.1.1		



Task No:	5.6.1	Partner:	EPFL
Function No:	5.6.1.3	Leader:	Sean Hill
Function Name:	Automated data monitoring pipeline		
Test Case:	Data uploaded in the DataSpace are automatically integrated and made available for querying.		
Planned Start Date:	January 2015	Planned Completion Date:	October 2015
Requires Functions:	5.6.1.2, T5.1.1 (shared data space)		

Task No:	5.6.1	Partner:	EPFL
Function No:	5.6.1.4	Leader:	Sean Hill
Function Name:	Data access API		
Test Case:	A user can specify a detailed query and obtains the relevant data in the format of choice (e.g. JSON, csv files).		
Planned Start Date:	October 2014	Planned Completion Date:	March 2015
Requires Functions:	5.6.1.2, 5.6.1.3		

Task No:	5.1.2	Partner:	EPFL
Function No:	5.1.2.1	Leader:	Sean Hill
Function Name:	Data mining		
Test Case:	Data are analysed by pre-defined sets of tools and a series of parameters are generated that can be used in simulations.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	5.6.1.4, 5.6.1.1		

Task No:	5.1.2	Partner:	EPFL
Function No:	5.1.2.2	Leader:	Sean Hill
Function Name:	Integrate data mining results		
Test Case:	Successful population of the database with results derived from analysis and linked to the appropriate ontologies. The new information is available for querying.		
Planned Start Date:	May 2015	Planned Completion Date:	September 2015
Requires Functions:	5.1.2.1		

Task No:	5.1.6	Partner:	EPFL
Function No:	5.1.6.1	Leader:	Sean Hill
Function Name:	Brain atlas navigator		
Test Case:	A user is able to navigate through the brain in different planes using a web browser. The navigator will allow the user to create a brain atlas from a stack of high-resolution 2D images.		
Planned Start Date:	May 2014	Planned Completion Date:	April 2015
Requires Functions:			

Task No:	5.1.6	Partner:	EPFL
Function No:	5.1.6.2	Leader:	Sean Hill
Function Name:	Annotation editor		
Test Case:	A user will be able to annotate specific 2D images within the atlas. The annotations are searchable.		
Planned Start Date:	May 2014	Planned Completion Date:	September 2015
Requires Functions:	5.1.6.1		

2.2 Brain Atlas Builder (T5.1.6)

The Brain Atlas Builder is a web application for constructing brain atlases, and sharing them online.

2.2.1 Brain Atlas Builder: Overall Goals

The goal is to create queryable (including 3D spatial and metadata queries) brain atlases data storage.

The software will allow users to organise, query, access and integrate spatially and semantically tagged brain data; to configure the dimensions of the brain; to define its segmentation into structures (areas, regions, tracts, ventricles etc.); to assign names to the structures using terms from standard ontologies; and to connect spatial locations in the atlas with the models and literature in the HBP data space and the HBP Knowledge Graph. The software will be developed generically, ensuring it can be used both for mouse (and other rodent models) and human atlases.

2.2.2 Brain Atlas Builder: Use Cases

2.2.2.1 Create a brain atlas

Use Case ID:

- SP5-UC-011

Primary Actor:



- One Biological Scientific User Daniel

Success Scenario:

- Daniel would like to create a brain atlas from a stack of high-resolution 2D images located on his computer, and share the brain atlas online.
- Daniel creates a project - Project1. He is now the owner of Project1.
- Daniel uploads 2D images to Project1.
- Daniel is requested to enter the metadata (HBP-CORE or PAB).
- Daniel starts the converter via Web GUI. The converter creates pyramids, previews and thumbnails from the 2D images. This might be a time consuming task, Daniel might log out and login later to check the progress.
- The conversion is finished. Now the following extra tools and features are automatically available via unique URLs for Daniel only:
 - Original 2D images, pyramids, thumbnails and previews.
 - 2D image viewers for individual 2D images in full resolution (zoom and pan).
 - 2D filmstrip viewers for a set of 2D images in full resolution (select image, zoom and pan).
 - Annotation editor for individual 2D images in full resolution (draw an annotation, zoom and pan).
- Via Web GUI Daniel selects which 3D template package he wants to anchor to and the 2D image to anchor.
- Daniel uses AlignNII, allowing a step-wise process to anchor the 2D images to the 3D template package. The steps include:
 - Identification of the orientation of the 2D image stack in relation to the 3D template package.
 - Global scaling and positioning of the individual images (3D best fit) to the 3D template package.
 - Local positioning (region - by - region) of regions of interest.
- Daniel saves the anchoring results.
- Now the 2D images are searchable using metadata and/or 3D spatial requests, and can be viewed individually, in filmstrips or in 3D viewer.
- Daniel can add other users to Project1 and give them read or read-write access OR make Project1 publicly (anonymously) read-only available. The tools inherit the access rights.
- Optional: the tools, original 2D images, pyramids, thumbnails and previews can be embedded into web projects.

2.2.2.2 Search by 3D spatial location or metadata

Use Case ID:

- SP5-UC-012

Primary Actor:

- One Biological Scientific User John

**Success scenario:**

- John would like to find all available brain atlases and/or 2D images for a particular 3D location and/or a particular set of metadata.
- John logs in to the Collaboratory, defines the criteria and runs the search.
- The results can be displayed individually, in filmstrips or in 3D viewer.

2.2.2.3 Annotate brain atlas**Use Case ID:**

- SP5-UC-013

Primary Actors:

- Two Biological Scientific Users: Bill and Daniel

Success Scenario:

- Bill would like to highlight certain areas on 2D images of the brain atlas created by Daniel (owner) as Project1.
- Daniel gives read-write access to Bill for Project1.
- Bill selects a 2D image and opens Annotation editor for it.
- Bill draws an annotation, saves it and repeats the process for every 2D image he wants to annotate.
- The 2D images with the annotations can be displayed individually or in filmstrips. The annotations can also be displayed in 3D viewer as 3D meshes. The annotations are searchable.

2.2.3 Brain Atlas Builder: Functional Requirements**2.2.3.1 Authentication and authorisation****Functional Requirement ID:**

- SP5-FR-012

Features:

- Satisfies all Functional Requirements for Authorisation and Authentication of the Collaboratory.

Dependencies:

- Relying on the SSO Authentication and Authorisation functionality of the Collaboratory.

2.2.3.2 3D template packages**Functional Requirement ID:**

- SP5-FR-013

Dependencies:

The component requires at least one 3D template package. Currently available packages:

- Waxholm Rat.
- ABA Mouse.

2.2.4 Brain Atlas Builder: Non-Functional Requirements

2.2.4.1 Efficiency

The tools of the component are client-server based. While the downloading of an entire data set is possible, the Imagery service feeds the minimal amount of data necessary for the tools to operate effectively saving the bandwidth.

2.2.4.2 Reliability

The Imagery service and the tools were successfully tested on average bandwidth line (1Mbit/s) with 2D data sets with over 100000x100000 pixels dimensions and 3D data sets with over 1024x512x512 voxels dimensions.

2.2.5 Brain Atlas Builder: Software

The 3D Brain Atlas Builder will be provided as a set of web services and linked to the Atlas Embedding Component described in the Brain Simulation Platform.

2.2.6 Brain Atlas Builder: Physical Architecture

2.2.6.1 Components

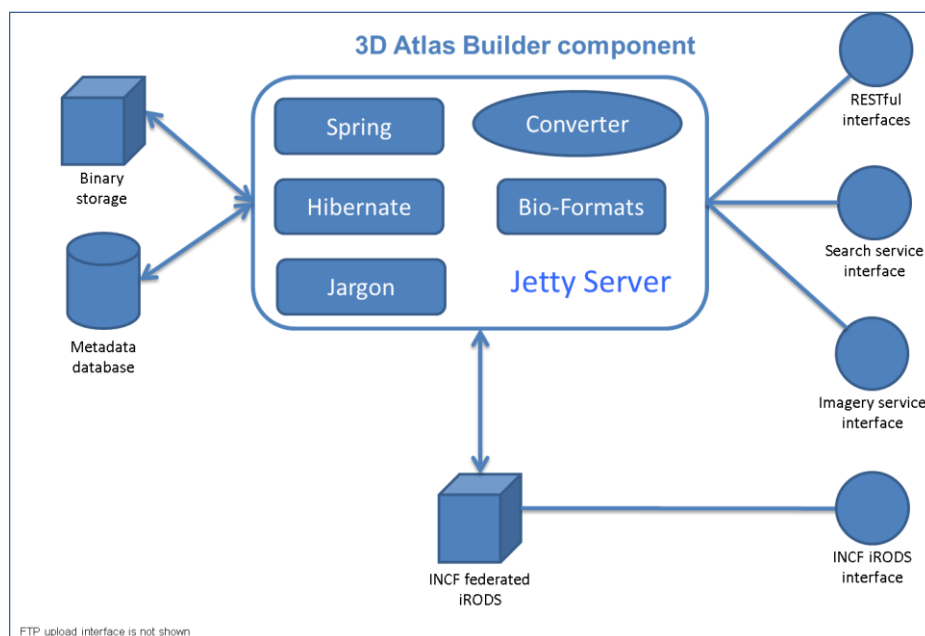


Figure 9: Brain Atlas Builder

AlignII tool will use RESTful and Imagery interfaces.

2.2.7 Brain Atlas Builder: Services required from other Platforms

None.



2.2.8 Brain Atlas Builder: Services provided to other Platforms

- Imagery service:
 - Pyramid's tiles for 2D viewers.
 - Standard 3 plane cut of 3D data set (NIfTI): Waxholm, ABA.
 - Arbitrary cut of 3D data set (NIfTI): Waxholm, ABA.
- Metadata and 3D spatial search.
- RESTful interfaces to the major operations.
- The service can be registered at Collaboratory.

2.2.9 Brain Atlas Builder: Prerequisites

2.2.9.1 Required

- NORSTORE - UIO and national infrastructure; provides iRODS hosting and maintenance.
- USIT - UIO computing services; provides hosting, backup and maintenance for the metadata database (Oracle) and the disk storage (up to 80TB DAS).

2.2.9.2 Preferred

None.

2.2.10 Brain Atlas Builder: Necessary Parallel Activities

Access to Allen Brain Institute 3D brain meshes and ontologies will be essential to populating and configuring the 3D Brain Atlas builder framework for the Allen Mouse Atlas.



3. Tools for Structural Data Analysis (WP5.2 and WP5.4)

Much of the structural data produced by modern neuroscience takes the form of image stacks from light and electron microscopy, MRI, PET etc. Given that many of these techniques produce terabytes of data in a single session, the best way to unlock the information they contain is through automatic image processing. The HBP will develop tools for this purpose (T5.2.1, T5.2.2), which it will share with the community via the INCF. The tools will include software to automate the extraction of cell densities and distributions; the reconstruction of neuron morphologies; the determination of subcellular properties such as synapse and organelle geometry, size and location; and the identification of the long- range fibre tracts underlying connectivity.

This chapter covers the following specific tools:

- Analysis of experimental data obtained by EM and light microscopy (EMDigest).
- Segmentation and annotation of brain tissue microscopy stacks (EspINA).
- Analysis of synapse distribution (3DSynapsesSA).
- Reconstruction of neuron morphology of neurons (3DPyrStructure).
- Reconstruction of soma morphology (3DSomaMS).

3.1 Structural Tools: EMDigest (T5.2.1)

This section is present the specification of the EMDigest Component of the NIP.

3.1.1 *EMDigest: Overall Goals*

The EMDigest component is a tool providing interaction functionalities for improving the interactive analysis of experimental data obtained by EM and light microscopy.

With EMDigest, users will be able to perform interactive filtering of experimental EM data, visualising and interacting with the obtained results in a user-driven exploratory navigation and analysis process.

EMDigest is part of the Filter and Compare component developed in collaboration with WP7.3 (HPC: Visualisation). The data navigation and analysis features will be exploited for the use with both experimental (light and EM microscopy images) and simulation data.

3.1.2 *EMDigest: Use Cases*

3.1.2.1 Sample interactive analysis session

Use Case ID:

- SP5-UC-014

Primary Actor:

- One Scientific User (Ruth)

**Preconditions:**

- A series of segmented structures organised as different sets, depending on the specimen or tissue sample they come from. These segmented structures should be annotated according to the hierarchy of concepts of a given ontology.

Success Scenario:

- 1) Ruth would like to study a set of volumetric images that contains a population of segmented morphologies of interest (e.g. dendritic spines), trying to identify outliers in the population.
- 2) Ruth will load one or multiple sets of segmented structures.
- 3) Ruth will decide if she works with all the morphologies or will select just a subset according to categorical criteria (e.g. basal vs. apical dendrites).
- 4) Then she will filter the selected subset to extract the smallest spines according to the interactive threshold that she fixes with a slider.
- 5) Maybe she wants to compare this subset with another subset extracted in a similar way. She will be able to compare both subsets using population charts (e.g. box plots) and statistical tests (e.g., t-test).
- 6) When Ruth is happy with the comparisons made and decide to finish the working session, she can export the results to standard formats.

3.1.3 EMDigest: Functional Requirements**3.1.3.1 Authentication and authorisation****Functional Requirement ID:**

- SP5-FR-014

Features:

- No specific authentication function.

3.1.3.2 Data references**Functional Requirement ID:**

- SP5-FR-015

Features:

- EMDigest will operate on local data or using remote data stored in a common repository.

3.1.3.3 Filter and compare service**Functional Requirement ID:**

- SP5-FR-016

Features:

There is a list of common requirements with the rest of the “Filter and Compare” suite of tools jointly developed between WP5.2 and WP7.3.

- 1) The filter service must provide a selection of morphological structures extracted from the population.



- 2) Morphological data available in the subsets selected must be exported to spread sheet format.
- 3) The user must be able to answer questions like: Are these subsets belonging to the same distribution or follow the same pattern? The structures of subset A are smaller/greater than the structures of the subset B?
- 4) The compare service must provide statistical information about selected subsets (e.g., t-tests).
- 5) This service will present users visual representations (e.g., box plots) about different morphological parameters of two subsets.
- 6) The component will support the inclusion of new plug-ins (e.g., 3D navigation).

3.1.4 EMDigest: Non-Functional Requirements

3.1.4.1 Interfaces

- Specification of interactions must be done with the user-centred design methodology:
 - Interfaces must be as simple as possible to reduce the learning curve.
 - Interfaces must adapt to the scientists methodology, not the other way.
- Major functions must be accessible from both Web GUI and programmatic Web Service clients.

3.1.4.2 Efficiency

EMDigest will rely on the performance of the Filter and Compare service at the HPC Platform (Visualisation).

3.1.4.3 Reliability

EMDigest will rely on the availability of the Filter and Compare service at the HPC Platform (Visualisation).

3.1.5 EMDigest: Software

- Core module: integrates data types, data operations and interaction state.
- InfoVis Views: provides information visualisation interactions according to the tasks and data involved in the use case described.
- Statistics Module: provides statistical tests about the population under study.
- IO Module: import/exports data and results from/to standard formats.

3.1.6 EMDigest: Physical Architecture

We refer to the HPC / Visualisation architecture of the Filter and Compare component.

3.1.7 EMDigest: Services required from Other Platforms

- DataSpace - Neuroinformatics - EMDigest accesses DataSpace stored data.



- EspINA - Neuroinformatics - EMDigest uses EspINA API to visualise data stored in native format.

3.1.8 EMDigest: Prerequisites

3.1.8.1 Required

No particular requirements.

3.1.8.2 Preferred

Although EMDigest may work autonomously, we prefer that a DataSpace - Neuroinformatics repository support it.

3.1.9 EMDigest: Necessary Parallel Activities

- EMDigest will require integration support with the DataSpace platform.
- EMDigest will require aligning the analysis concepts with the KnowledgeSpace ontologies.

3.2 Structural Tools: EspINA (T5.2.2)

This section presents the specification of the EspINA Component of the NIP.

3.2.1 EspINA: Overall Goals

The EspINA component is an image segmentation and annotation tool to process light and electron microscopy stacks of brain tissue. This tool provides a wide range of image segmentation techniques (depending on the corresponding structure to identify) from fully manual or assisted mechanisms to automatic and semi-automatic techniques.

The EspINA component is designed to work on the images produced by the SP1 experimentalists, although it could be extended to use with other images. The main goal is to provide a high-throughput efficient interaction with the kind of data produced by the HBP (in EM/light microscopy). This goal has a major influence on how specific image segmentation algorithms are tailored and optimised.

The design of EspINA follows a user-centred paradigm to grant a fast learning curve, and to support the interaction and operation requirements proposed by the domain experts (final users).

3.2.2 EspINA: Use Cases

3.2.2.1 Assisted neuron structure segmentation

Use Case ID:

- SP5-UC-015

Primary Actor:

- One Scientific User (Angel)

Preconditions:



- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.

Success Scenario:

- 1) Angel has identified a structure that needs to be segmented which can be automatically identified by one of the tools provided by EspINA.
- 2) He opens EspINA and adds the stack to the analysis. EspINA requests, if needed, the meta-information to identify the sample which it was obtained from and lets the him modify it if necessary.
- 3) Once the stack is loaded, it is displayed using one slicing view for each orthogonal plane.
- 4) Then, he selects the appropriate tool on the toolbar to identify the neuron structure.
- 5) The tool may require some parameters to operate, providing default values for all of them.
- 6) He uses the tool to identify all the structures on the stack and once it has finished, he decides which ones he wants to keep.

3.2.2.2 Semi-automated neuron structure segmentation**Use Case ID:**

- SP5-UC-016

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.

Success Scenario:

- 1) Angel has identified a structure that needs to be segmented which cannot be automatically identified by one of the tools provided by EspINA.
- 2) He opens EspINA and adds the stack to the analysis. EspINA requests, if needed, the meta-information to identify the sample from which it was obtained, and allows him to modify it if necessary.
- 3) Once the stack is loaded, it is displayed using one slicing view for each orthogonal plane.
- 4) Angel identifies a new structure to be segmented using the slice views.
- 5) Then, he selects the appropriate tool (i.e. Seed Grow Segmentation tool) on the toolbar to identify the neuron structure.
- 6) Angel chooses from the classification the category that fits the segmentation he wants to obtain.
- 7) The tool may require some parameters to operate, providing default values, which will be set depending on the neurological category he had already selected, for all of them.
- 8) He uses the tool to indicate in one of the views the location of the structure.
- 9) EspINA will start reconstructing a new segmentation with all the voxels with the same grey levels than the selected voxel.



10) Once it has finished it is displayed in the segmentation explorer as well as in all the views.

11) He repeats this process until all the structures have been identified and segmented.

3.2.2.3 Manual neuron structure segmentation

Use Case ID:

- SP5-UC-017

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.

Success Scenario:

- 1) Angel has identified a structure that needs to be segmented, but the automated tools cannot obtain a correct morphology.
- 2) Angel has identified a structure that needs to be segmented, which cannot be automatically or semi-automatically identified by one of the tools provided by EspINA.
- 3) He opens EspINA and adds the stack to the analysis. EspINA requests, if needed, the meta-information to identify the sample from which it was obtained, and allows him to modify it if necessary.
- 4) Once the stack is loaded, it is displayed using one slicing view for each orthogonal plane.
- 5) Angel identifies a new structure to be segmented using the slice views.
- 6) Then, he selects the appropriate tool (i.e. brush tool) on the toolbar to manually create the neuron structure.
- 7) The tool may require some parameters to operate, providing default values, which will be set depending on the neurological category he had already selected, for all of them.
- 8) He uses the tool to mark on the views the voxels belonging to the structure.
- 9) After the first voxel is annotated a new segmentation is displayed in the segmentation explorer as well as in all the views.
- 10) He repeats this process until all the structures have been identified and segmented.

3.2.2.4 Manual edition of segmented structure

Use Case ID:

- SP5-UC-018

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.



- The 3D stack will also reference a segmentation containing one or more segmented structures.

Success Scenario:

- 1) Angel is not satisfied with a segmentation obtained by any of the EspINA tools.
- 2) He decides to manually edit the segmentation voxels.
- 3) He selects the segmentation and the edition tool that best fits its needs (brush, close, open, erode, dilate, planar split, merge or subtract).
- 4) Once he has made the modifications to the segmentation, it has the morphological appearance that he desires and is ready for a later analysis.

3.2.2.5 Synaptic apposition surface (SAS) analysis**Use Case ID:**

- SP5-UC-019

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible form a central repository.
- The 3D stack will also reference a segmentation containing one or more segmented structures. These structures should be the post synaptic densities.

Success Scenario:

- 1) Angel has segmented a number of synapsis with EspINA and wants to obtain the information relative to their apposition surfaces.
- 2) He selects the synapsis he is interested in using the views and/or the segmentation explorer.
- 3) Then he uses the automatic apposition surface extractor tool to generate the apposition surface of each selected synapse.
- 4) Once the tool has finished its execution he selects the resulting apposition surfaces and opens the apposition surface analysis dialogue, where he can see all the apposition surface information of every synapse.

3.2.2.6 Analysis revision**Use Case ID:**

- SP5-UC-020

Primary Actor:

- Two Scientific Users (Angel and Javier)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible form a central repository.



- The 3D stack will also reference a segmentation containing one or more segmented structures. These structures should be the post synaptic densities.

Success Scenario:

- 1) Once Angel has identified all the structures needed for his analysis, he shares the resulting segmentation file to his colleague, Javier.
- 2) Javier reviews all the segmentations and in case of discordance he tags those segmentations as doubtful.
- 3) In case he finds a segmented structure that Angel has missed, he adds it to the analysis himself using the segmentation tools.
- 4) When Javier has finished, they discuss their opinions and finally they classify each segmented structure accordingly.

3.2.2.7 Training

Use Case ID:

- SP5-UC-021

Primary Actor:

- One Scientific User (Angel) and one scientific student (Rose)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible form a central repository.

Success Scenario:

- 1) Rose is learning to segment neural structures using EspINA.
- 2) After a briefing from her teacher, Angel, she starts identifying all the structures of the stack.
- 3) When she has finished, she sends the analysis to Angel for evaluation.
- 4) Angel reviews and adds tags or notes to the analysis where he thinks it is convenient.
- 5) He then sends the analysis file back to Rose.
- 6) Rose then opens the analysis and inspects the corrections and annotations Angel has made.

3.2.2.8 Collaborative segmentation

Use Case ID:

- SP5-UC-022

Primary Actor:

- Two Scientific Users (Angel and Rodrigo)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible form a central repository.

Success Scenario:



- 1) Angel decides to split his workload in the segmentation of the structures of a stack with his colleague Rodrigo.
- 2) Each of them segments one half of the stack.
- 3) When they have finished, they gather and open both analyses at the same time, so they get the segmentations of all the structures of the stack.

3.2.2.9 Data extraction

Use Case ID:

- SP5-UC-023

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.
- The 3D stack will also reference a segmentation containing one or more segmented structures.

Success Scenario:

- 1) Angel has to extract some of the information contained within an EspINA file.
- 2) He starts EspINA in his machine and opens the corresponding file.
- 3) Angel then selects the relevant segmentations that the information will be extracted from.
- 4) He then opens the segmentation information dialogue showing the representations and information of the relevant segmentations.
- 5) He selects the properties he wants to extract, excluding all the information not relevant at the moment.
- 6) Then he is asked to enter a file name and a format in which the information will be saved.
- 7) EspINA extracts only the relevant information and saves it to disk.

3.2.2.10 Import data from previous versions

Use Case ID:

- SP5-UC-024

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.
- The 3D stack will also reference a segmentation containing one or more segmented structures. This stack has been previously segmented with another version of the tool.

**Success Scenario:**

- 1) Angel needs to verify some data generated with an older version of EspINA.
- 2) He starts EspINA in his machine and opens the relevant file.
- 3) EspINA imports all the relevant data contained in the old file format and starts a new session with the data.
- 4) Angel can now inspect the data, modify or generate new data if necessary and save the results in a new EspINA analysis.

3.2.2.11 Export view snapshots and 3D scene**Use Case ID:**

- SP5-UC-025

Primary Actor:

- One Scientific User (Angel)

Preconditions:

- The user should have access to the corresponding 3D stack of images, either as a local file or accessible from a central repository.
- The 3D stack will also reference a segmentation containing one or more segmented structures.

Success Scenario:

- 1) Angel has completed an experiment with EspINA in his machine and needs some images for a paper publication.
- 2) He starts EspINA and loads the segmentation file containing the relevant information of the experiment.
- 3) He configures the views until he decides he has the right configuration from which he wants to get images.
- 4) Angel uses the 3D export tool to export several 2D images from the 3D view.
- 5) He also wants to export the scene to a Blender-compatible format, so he uses the 3D export tool to obtain a VRML description of the scene he will later use to obtain additional images using an external tool.

3.2.3 EspINA: Functional Requirements**3.2.3.1 Authentication and authorisation****Functional Requirement ID:**

- SP5-FR-017

Features:

- EspINA is provided as a standalone tool downloadable from Neuroinformatics or HBP Platforms websites.

Dependencies:

- The access to any common repository via EspINA is subjected to any of the regular access protocols defined by the HBP.



3.2.3.2 Data references

Functional Requirement ID:

- SP5-FR-018

Features:

- EspINA will operate on local data or using remote data stored in a common repository.

3.2.3.3 Load microscopy images

Functional Requirement ID:

- SP5-FR-019

Features:

- EspINA must open stack of images produced in a multi-image TIFF file format (the one provided by the crossbeam EM microscope, but also supported by other microscopy equipment).
- Sometimes images need some registration and pre-processing. In these cases the images may come in meta-image format (mhd/mha).

3.2.3.4 Load/store segmentation data

Functional Requirement ID:

- SP5-FR-020

Features:

- There is different information produced by EspINA tool that has to be stored and loaded for posterior analysis.
 - Previous versions of EspINA used a variation of the meta-image format to store segmentation data along with its metadata. EspINA must be compatible with the previous format (segmha).
 - EspINA defines its own file format (.seg) to store analysis information.

3.2.3.5 Collaborative segmentation

Functional Requirement ID:

- SP5-FR-021

Features:

- Load two analyses of the same stack, or different ones (tiling) in the same session.

3.2.3.6 Segmentation tools

Functional Requirement ID:

- SP5-FR-022

Features:

- EspINA must provide tools to segment the images allowing the specification of a Region Of Interest (ROI) that will limit the segmentation algorithm.
- The user can choose on which stacks the segmentation tools will be applied.



- Several tools for manual, semi-automatic and assisted segmentation are provided:
 - Growing seed segmentation, allowing the user to use specify the best pixel for the seed origin.
 - Manual segmentation using a brush defined by a geometric area or volume.
 - Contour tool that allows the user to define an area that delimits the segmentation.
 - Assisted techniques to identify subcellular structures (EM images): mitochondria, myelin and post-synaptic density.
 - Assisted techniques to identify cells and larger structures (light microscopy images): cell nuclei (neurons and glial cells) and blood vessels.
- The modification of existing segmentations can be done using the edition tools:
 - Manual segmentation tools (also used for segmentation creation, that is, brushes and contour tools).
 - Morphological modification tools that use morphological algorithms to obtain certain morphological qualities of the segmentation (erode, dilate, open and close algorithms).
 - Boolean operation tools (add and subtract segmentations with and from each other).
 - Hole-filling algorithms.
 - Split tool to divide a segmentation using an oriented infinite plane.

3.2.3.7 Segmentation classification

Functional Requirement ID:

- SP5-FR-023

Features:

- Annotate segmentations using hierarchical categories.
- Category classification will need to be editable and extendible by the user.

3.2.3.8 Segmentation inspection

Functional Requirement ID:

- SP5-FR-024

Features:

- Users can view all the relevant information of a given segmentation
 - Graphical 2D/3D representation
 - Parameters of creation (depending on the algorithm/tool used for creation).
 - Additional information associated to the segmentation and provided by third party extensions.
- Export relevant information to common formats (Excel and Comma-Separated Values formats) to easily exchange information with external tools.



3.2.3.9 Stereological counting frame definition

Functional Requirement ID:

- SP5-FR-025

Features:

- The software allows the user to define stereological counting frames that can be:
 - Adaptive
 - Orthogonal
- Stereological counting frames can be automatically adjusted to exclude the segmentation that may be incomplete by moving the inclusion margin.
- Stereological counting frames can take into consideration a category or group of categories to be applied on (classification criteria).
- Stereological counting frames can be modified manually by editing its margins or modifying them interactively on the different 2D views.
- Stereological counting frame information can be exported.

3.2.3.10 Synaptic apposition surface

Functional Requirement ID:

- SP5-FR-026

Features:

- SAS can be generated for segmentations categorised as synapses and visualised in 2D or 3D.
- SAS metrics are computed and available for data analysis.
- Provides a special report that associates synopsis measures with its corresponding SAS measures.

3.2.3.11 Neuron structure data

Functional Requirement ID:

- SP5-FR-027

Features:

- Once the user has finished segmenting neuron structures, he will need to obtain different types of data of each structure
 - Morphological information, such as centroid, size or Feret diameter, is provided.
 - Distance to the edges of the stacks (may determine whether a segmentation is potentially incomplete).
 - Due to segmentation incompleteness caused by sampling, stereological information will help users to decide whether to take or not into account the information obtained by the tool.
 - Sometimes the user will need to annotate segmentations with text notes to describe useful information.
 - Tags are also necessary to ease classification of segmented structures.



3.2.3.12 Neuron structure analysis

Functional Requirement ID:

- SP5-FR-028

Features:

- Tabular report of information extracted from segmentations.
- Can be exported as csv or xls.
- Information to be displayed can be configured.

3.2.3.13 Stack visualisation

Functional Requirement ID:

- SP5-FR-029

Features:

- Sometimes users will need to calibrate the size of each voxel dimension to properly visualise the stack of images.
- The software allows the modification of the visual representation of the stacks giving the possibility of changing the contrast, brightness, opacity and stain of a stack or a group of stacks.
- It is possible to load several stacks into a session.

3.2.3.14 Interactive visualisation

Functional Requirement ID:

- SP5-FR-030

Features:

- Change channel visibility.
- Change crosshair visibility.
- Toggle all segmentations visibility with a shortcut.
- Change segmentation visibility.
- Choose graphical representation for each item (channel/segmentation).
- Colour segmentation using different criteria:
 - By Segmentation Category / Annotation concept.
 - By number.
 - By Stereological Counting Frame state.
- The segmentations can be grouped together:
 - By sample.
 - By category.



3.2.3.15 User interaction

Functional Requirement ID:

- SP5-FR-031

Features:

- Every operation done in the software can be undone and redone if necessary.
- Some visualisation tools allows the user to modify the views to his/her needs by hiding segmentations or channels, zooming over a region of interest (zoom tool) or highlighting the region where the 2D views cross each other (crosshair).
- Contextual menus are provided in key views of the software allowing the user to modify the selection or the visualisation of the representations.
- Time consuming operations can be interrupted.

3.2.3.16 Export scenes to common 3D formats

Functional Requirement ID:

- SP5-FR-032

Features:

- In order to create additional material with external tools, users will need to export current 3D scene to formats supported by Blender, such as VRML or X3D.

3.2.4 *EspINA: Non-Functional Requirements*

3.2.4.1 Framework

- Information provided by segmentations and channels must be extensible by plugins.
- Graphical representations for segmentations and channels must be extensible by plugins.
- New tools or panels can be added by plugins.
- Different colouring techniques may be added by plugins.

3.2.4.2 Graphical User Interfaces

- Specification of interactions must be done with the user-centred design methodology:
 - Interfaces must be as simple as possible to reduce the learning curve.
 - Interfaces must adapt to the scientists' methodology, not the other way around.

3.2.4.3 Efficiency

- Low latency is crucial for slicing exploration.
- Due to the large amount of data, reducing memory footprint is required.
- Exploit the concurrency of high-performance computing in intensive computing tasks.

3.2.4.4 Reliability

- EspINA must be robust to any input produced by the user.



- Tools will need to enforce good use of them.
- In case of crash, a recovery mechanism is needed.

3.2.4.5 Documentation

- User manual.
- API documentation for external plugins developers.

3.2.4.6 Multi-platform

- Linux.
- Windows.
- Mac.

3.2.5 *EspINA: Software*

- Core module: integrates data types, data operations and analysis state.
- Filters module: provides basic operations for data filtering, feature extraction, counting procedures, etc.
- Extensions module: extends dynamically basic data types to provide extra features on demand.
- GUI module: provides components needed to create graphical user interfaces that display and interact with core data structures.
- Support module: inter-connects different modules of the EspINA application and is used to develop new plugins.
- Application module: packages all the framework components into a single application that can be extended, and that is designed to satisfy final user requirements.
- Stereological Information Plugin module: defines different types of stereological counting frames and provides stereological information for every segmentation.
- SegmhalImporter Plugin module: extends EspINA to load the initial version file format.
- Synaptic Apposition Surface Plugin module: extends EspINA to extract SAS for synapsis.
- Regularised Automatic Autosegmenter Plugin module: automatic segmentation tool based on an iterative training provided by the user.

3.2.6 *EspINA: Physical Architecture*

EspINA is supported by any local storage or remote repository of experimental data (including raw images and produced segmentation files).

3.2.7 *EspINA: Services required from other Platforms*

- DataSpace - Neuroinformatics - Export analysis data in the format specified by the DataSpace - Neuroinformatics - Use EspINA API to populate the DB with data stored in EspINA (.seg) format.
- EMDigest - Neuroinformatics - Use EspINA API to visualise data stored in native format.



- Filter and Compare - HPC/Visualisation - Use EspINA API to visualise data stored in native format.

3.2.8 *EspINA: Services provided to other Platforms*

DataSpace - Neuroinformatics - EspINA will provide enriched data and annotated segmentation to the DataSpace repository.

3.2.9 *EspINA: Prerequisites*

3.2.9.1 Required

- No particular requirements

3.2.9.2 Preferred

- Although EspINA may work autonomously, it preferable to have it supported by a DataSpace - Neuroinformatics repository.

3.2.10 *EspINA: Necessary Parallel Activities*

- EspINA will require integration support with the DataSpace platform.
- EspINA would require aligning the analysis concepts with the KnowledgeSpace ontologies.

3.3 Structural Tools: 3DSynapsesSA (T5.4.2)

The 3DSynapsesSA component is designed to perform spatial analysis of synapses in three dimensions.

3.3.1 *3DSynapsesSA: Overall Goals*

3DSynapsesSA enables the analysis of the three-dimensional spatial distribution of a set of synapses; particularly it allows one to check whether the distribution is random (complete spatial randomness, CSR) or whether it follows a random sequential adsorption (RSA) process. The component also allows the user to visualise the synapses and to perform different summary graphs of their distribution.

3.3.2 *3DSynapsesSA: Use Cases*

Use Case ID:

- SP5-UC-033

Primary Actor:

- One Scientific User, Daniel

Success Scenario:

- Scientific Developer User Daniel would like to analyse the three-dimensional spatial distribution of a set of synapses.



- Daniel needs to use the R program and load some necessary packages (the *spatstat* package is essential but some others can be needed for certain processing operations).
- Daniel loads the (x, y, z) coordinates of each synapse and, if he has this information, its diameter.
- Daniel analyses whether the pattern defined by synapses coordinates follows a CSR process. To do this he can choose between different statistical tests.
- He runs three-dimensional CSR processes simulations with the same characteristics of the synapses he is studying and he compares the simulations with the real samples using statistical tests.
- If he has the synapses diameters, he can do the same as in the previous point but with RSA processes simulations.
- Daniel graphically represents G, F, K and L functions of simulations patterns and real samples.

3.3.3 3DSynapsesSA: Functional Requirements

3.3.3.1 Authentication and authorisation

Functional Requirement ID:

- SP5-FR-036

Features:

- The use of the R source and its packages is not constrained by an authentication or authorisation system.

3.3.3.2 Data analysis

Functional Requirement ID:

- SP5-FR-037

Features:

- It is necessary to have the (x,y,z) coordinates of the synapses and, additionally, their diameters, in one or several files.
- After loading the data, the component will be able to visualise the synapses in three dimensions, adjust different spatial models using statistical tests, and perform the most commonly used graphs in spatial statistics to analyse the data distribution.

3.3.4 3DSynapsesSA: Non-Functional Requirements

3.3.4.1 Interfaces

Functions must be accessible from R scripts. They can be executed from command prompt or an IDE.



3.3.4.2 Efficiency

Data handling needs to be compatible with common formats for data set representation to facilitate efficient use of storage and I/O resources.

3.3.5 3DSynapsesSA: Software

- Synapse spatial model builder.
- Synapse spatial model simulator.
- Synapse spatial model and simulation plotter.

3.3.6 3DSynapsesSA: Physical Architecture

This section describes the components that make up the 3DSynapsesSA.

3.3.6.1 Components

- Module for adjusting spatial models.
- Module for simulations.
- Module for graphical representations.

The first module will include different spatial point process models: complete spatial randomness and random sequential adsorption. The simulation module will be used for goodness-of-fit tests. Finally, the last module will provide the graphical representations of G, F, K and L functions of simulation patterns and real samples.

3.3.7 3DSynapsesSA: Services required from other Platforms

None.

3.3.8 3DSynapsesSA: Services provided to other Platforms

Analysis and simulations of synapses distribution in three-dimensional space.

3.3.9 3DSynapsesSA: Prerequisites

3.3.9.1 Required

- R software environment.
- R spatstat package.
- R additional packages.

3.3.9.2 Preferred

None.

3.3.10 3DSynapsesSA: Necessary Parallel Activities

The 3DSynapsesSA needs (x,y,z) coordinates of the synapses and, additionally, their diameters, in common format that language R can handle. We need data of SP1 for synapses and diameters.

3.3.11 3DSynapsesSA: Hardware/Software Functions

This section describes a set of modules, with examples of the key functions they will contain which will be provided by the toolbox.

3.3.11.1 3DSynapsesSA: Hardware/Software Functions - Synapse Spatial Model Builder

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.1	Leader:	Pedro Larranaga
Function Name:	<i>Synapse spatial model builder</i>		
Test Case:	A user analyses whether the pattern defined by synapses coordinates follows a complete spatial randomness (CSR) or a random sequential adsorption (RSA) process.		
Planned Start Date:	March 2014	Planned Completion Date:	June 2014
Requires Functions:			

3.3.11.2 3DSynapsesSA: Hardware/Software Functions - Synapse Spatial Model Simulator

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.2	Leader:	Pedro Larranaga
Function Name:	<i>Synapse spatial model simulator</i>		
Test Case:	A user runs three-dimensional CSR or RSA processes simulations with the same characteristics of the synapses he is studying.		
Planned Start Date:	July 2014	Planned Completion Date:	Oct 2014
Requires Functions:	5.4.2.1		

3.3.11.3 3DSynapsesSA: Hardware/Software Functions - Synapse spatial model and simulation plotter

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.3	Leader:	Pedro Larranaga
Function Name:	<i>Synapse spatial model and simulation plotter</i>		
Test Case:	A user compares the simulations with the real samples using statistical tests and graphically represents G, F, K and L functions of simulations patterns and real samples.		
Planned Start Date:	Nov 2014	Planned Completion Date:	Feb 2015



Requires Functions:	5.4.2.1, 5.4.2.2
---------------------	------------------

3.4 Structural Tools: 3DPyrStructure (T5.4.2)

The component 3DPyrStructure is designed to perform Spatial Structural Analysis of Dendritic Pyramidal Neurons in 3D.

3.4.1 3DPyrStructure: Overall Goals

The 3DPyrStructure is intended to analyse and reconstruct a part of a neuron, the basal dendrite arrangement and its particular dendritic trees. A hybrid Bayesian network will model the different kinds of morphological variables and their relationships explaining the probabilistic dependences between them.

3.4.2 3DPyrStructure: Use Cases

Use Case ID:

- SP5-UC-034

Primary Actor:

- One Scientific User, Daniel

Success Scenario:

- Scientific Developer User Daniel has a set of neurons in NeuroLucida software format and he wants to analyse them in order to obtain a general descriptive analysis of the different neuron features.
- Daniel will change the NeuroLucida format (.DAT) to ASC using the windows command prompt (cmd.exe).
- Daniel will select the different features to be measured.
- Finally, the results will pop up.

Use Case ID:

- SP5-UC-035

Primary Actor:

- One Scientific User, Jane

Success Scenario:

- Scientific Developer User Jane has a set of neurons in NeuroLucida software format and she wants to get the conditional dependencies between the different features in order to get a fitted model.
- Jane will change the NeuroLucida format (.DAT) to ASC using the windows command prompt (cmd.exe).
- Jane will select the different features (variables) to be included.
- She will obtain the model that connects the selected variables. A graphical representation (directed acyclic graph) could also be displayed.



- Jane asks queries to the model, as a probability of some variables given other variables fixed to some values (evidence).
- Jane will receive these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph.

3.4.3 3DPyrStructure: Functional Requirements

3.4.3.1 Authentication and authorisation

Functional Requirement ID:

- SP5-FR-038

Features:

- Relying on the SSO Authentication and Authorisation functionality of the Collaboratory.
- Satisfies all Functional Requirements for Authorisation and Authentication of the Collaboratory.

3.4.3.2 Neuron selection

Functional Requirement ID:

- SP5-FR-039

Features:

- The layer(s) of neuron(s) can be selected by scripting.
- The selected neurons will be viewed in a data frame that will also contain some specific information about them, such as the number of dendrites per neuron.
- The selected subset could be exported to a specific format file, therefore an output path has to be provided.

3.4.3.3 Generation of descriptive analysis

Functional Requirement ID:

- SP5-FR-040

Features:

- The 3DPyrStructure will accept different types of variables to be analysed.
- The variables to be included in the descriptive analysis have to be selected from the selected subset of neurons.
- Descriptive analysis files will be generated from the input subset.
- The format and the path file to be saved could also be selected.

3.4.3.4 Bayesian network generation

Functional Requirement ID:

- SP5-FR-041

Features:



- A Bayesian Network will be generated from the selected subset of neurons (input data).
- In the graphical representation (directed acyclic graph) the nodes represent the variables and the arcs represent the conditional dependencies.
- The queries applied to the generated Bayesian Network will have to be scripted.

3.4.4 3DPyrStructure: Non-Functional Requirements

3.4.4.1 Interfaces

Functions must be accessible from R scripts. They can be executed from command prompt or an IDE.

3.4.4.2 Efficiency

Data handling needs to be compatible with common formats for data set representation and 3D representation to facilitate efficient use of storage and I/O resources.

3.4.4.3 Compatibility

R scripts should be multi-platform.

3.4.4.4 Maintenance

Source files must be commented to ease code interpretation.

3.4.5 3DPyrStructure: Software

- Pre-processing and exporting pyramidal neuron data.
- Basal pyramidal neuron feature extractor.
- Basal pyramidal neuron univariate analyser.
- Bayesian network builder of basal pyramidal neurons.
- Basal pyramidal neuron Bayesian network reasoner.

3.4.6 3DPyrStructure: Physical Architecture

This section describes the components that make up the 3DPyrStructure and how they interact with the other pieces of the NIP.

All the components are accessed through a network accessible service API. However, not all components will be exposed to end-users of the Neuroinformatics Portal. Some will be reserved for internal use.

3.4.6.1 Components

- Module for pre-processing and exporting the pre-processed files.
- Module for measuring the variable.
- Module for analysing the neurons.
- Module for generating the Bayesian network.

- Module for queries.

3.4.7 3DPyrStructure: Services required from other Platforms

- File format changing.
- Image processing and editing

3.4.8 3DPyrStructure: Services provided to other Platforms

- Dendritic basal arbour characterisation.
- Simulation of the dendritic basal arbour.
- Construction of a Bayesian network from neuron features.

3.4.9 3DPyrStructure: Prerequisites

3.4.9.1 Required

- Windows Command Prompt (CMD).
- R software environment.
- R packages.

3.4.9.2 Preferred

Neurolucida Explorer or NeuroMorphologyViewer.

3.4.10 3DPyrStructure: Necessary Parallel Activities

The 3DPyrStructure needs a set of pyramidal neurons in Neurolucida software format, provided by SP1.

3.4.11 3DPyrStructure: Hardware/Software Functions

This section describes a set of modules, with examples of the key functions they will contain which will be provided by the toolbox.

3.4.11.1 3DPyrStructure: Hardware/Software Functions - Pre-processing, exporting and feature extracting pyramidal neuron data

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.4	Leader:	Pedro Larranaga
Function Name:	<i>Pre-processing and exporting pyramidal neuron data</i>		
Test Case:	A user can perform basic checks in the data of pyramidal neurons and should change the Neurolucida format (.DAT) to R using the windows command prompt. Daniel will select the different features to be measured		
Planned Start Date:	March 2014	Planned Completion Date:	June 2014
Requires Functions:			



Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.5	Leader:	Pedro Larranaga
Function Name:	<i>Basal pyramidal neuron feature extractor</i>		
Test Case:	A user selects the different features to be measured and this function returns their values. Daniel will select the different features to be measured		
Planned Start Date:	July 2014	Planned Completion Date:	Oct 2014
Requires Functions:	5.4.2.4		

3.4.11.2 3DPyrStructure: Hardware/Software Functions - Basal pyramidal neuron univariate analyser

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.6	Leader:	Pedro Larranaga
Function Name:	<i>Basal pyramidal neuron univariate analyser</i>		
Test Case:	A user obtains with this function a univariate descriptive analysis of the morphological neuron measures.		
Planned Start Date:	Nov 2014	Planned Completion Date:	Feb 2015
Requires Functions:	5.4.2.4, 5.4.2.5		

3.4.11.3 3DPyrStructure: Hardware/Software Functions - Bayesian network builder of basal pyramidal neurons

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.7	Leader:	Pedro Larranaga
Function Name:	<i>Bayesian network builder of basal pyramidal neurons</i>		
Test Case:	A user obtains a Bayesian network model as a multivariate graphical representation (directed acyclic graph) whose nodes represent the morphological variables and the arcs the conditional probabilistic dependencies among them.		
Planned Start Date:	Mar 2015	Planned Completion Date:	Aug 2015
Requires Functions:	5.4.2.4, 5.4.2.5, 5.4.2.6		

3.4.11.4 3DPyrStructure: Hardware/Software Functions - Basal pyramidal neuron Bayesian network reasoner

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.8	Leader:	Pedro Larranaga
Function Name:	<i>Basal pyramidal neuron Bayesian network reasoner</i>		
Test Case:	A user asks queries to the model, i.e., the probability of some variables given other variables fixed to some values (evidence). He receives these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph. Jane will receive these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph.		
Planned Start Date:	Sep 2015	Planned Completion Date:	Dec 2015
Requires Functions:	5.4.2.4, 5.4.2.5, 5.4.2.6, 5.4.2.7		



3.5 Structural Tools: 3DSomaMS (T5.4.2)

The 3DSomaMS component is designed to characterise quantitatively a 3D soma, according to morphological features taken from its image reconstruction, to build a mathematical model that also will allow simulation of artificial somas.

3.5.1 3DSomaMS: Overall Goals

In the present component, we would like to statistically design the soma generation process. The first objective is to create an automatic typology of somas able to deal with the continuum of pyramidal soma morphologies. The second objective is the simulation from the previous model.

3.5.2 3DSomaMS: Use Cases

3.5.2.1 Quantitative characteristic extraction and soma typologies

Use Case ID:

- SP5-UC-036

Primary Actor:

- One Scientific User, Alex

Scenario:

- Alex has 3D data in his local machine corresponding to a representation of a soma.
- He wants to analyse soma's morphology and decided to download the R source with its dependencies (R packages).
- Alex considers interesting for his research to analyse the soma representation and he runs the script with the file as input.
- Alex supervises the input file for a correct pre-processing of the soma (cut of dendrites, filling holes, convex hull and smoothing).
- When pre-processing is finished Alex, selects a temporal path to save the pre-processed data and an additional path and name for the reconstructed mesh.
- Next a file path and name are requested to store the quantitative characteristics.
- Alex repeats the process iteratively for each soma to save data results in the output file and obtain a table with all of the soma characteristics.
- Finally an automatic typology of the soma shapes is obtained according to the cluster method selected by Alex.
- Alex can click on some group of somas to see its members as images.

3.5.2.2 Simulation of the model

Use Case ID:

- SP5-UC-037

Primary Actor:



- One Scientific User, Alex

Scenario:

- Once Alex has passed through the previous scenario he can perform this use case.
- Alex chooses a file with the soma characteristics and the model obtained from the clustering process. Alex starts the simulation of artificial somas of some prefixed typology.
- Alex has to set the parameters for the optimisation process, which transforms the simulated soma characteristics into a 3D soma image.
- He selects the path and file name to save the results.

3.5.3 3DSomaMS: Functional Requirements**3.5.3.1 Authentication and authorisation****Functional Requirement ID:**

- SP5-FR-042

Features:

- The use of the R source and its packages is not constrained by an authentication or authorisation system.

3.5.3.2 Selection of soma**Functional Requirement ID:**

- SP5-FR-043

Features:

- The system should read and process the somas selected by the user.
- The software should interpret VRML files.

3.5.3.3 3D representation**Functional Requirement ID:**

- 6) SP5-FR-044

Features:

- It is necessary to store soma data in your local machine.
- 3DSomaMS will need to have a mechanism to decide where to store output data.
- Data corresponding to 3D representation should be in a common format.
- 3D data should be displayed as a polyhedron.



3.5.3.4 Generate table with quantitative characterisation

Functional Requirement ID:

- SP5-FR-045

Features:

- Data stored is represented by variable names and soma ID.
- The file format should be in a common format.

3.5.3.5 Reconstruction of the soma

Functional Requirement ID:

- SP5-FR-046

Features:

- Soma must be pre-processed.
- Soma has to be repaired and reconstructed during pre-processing.
- Pre-processing can be supervised by the user.

3.5.3.6 Clustering

Functional Requirement ID:

- SP5-FR-047

Features:

- Different clustering methods should be provided.
- Individual somas within a group can be displayed after clustering.

3.5.4 *3DSomaMS: Non-Functional Requirements*

3.5.4.1 Interfaces

Functions must be accessible from R scripts. They can be executed from command prompt or an IDE.

3.5.4.2 Efficiency

Data handling must be compatible with common formats for data set representation and 3D representation, to facilitate efficient use of storage and I/O resources.

3.5.4.3 Compatibility

R scripts should be multi-platform.

3.5.4.4 Facility maintenance

Source files must be commented to ease code interpretation.

3.5.4.5 Monitoring

The user should monitor the soma pre-processing.



3.5.5 3DSomaMS: Software

- 3D pyramidal soma representation.
- Characterisation and typology of pyramidal somas.
- Pyramidal soma simulator.

3.5.6 3DSomaMS: Physical Architecture

This section describes the components that make up the 3DSomaMS.

3.5.6.1 Components

- Module for 3D representation and pre-processing.
- Module for characterisation and typology of somas.
- Module for simulation of the somas.

The module for characterisation and typology needs 3D pre-processing before its computation. The simulation module is executed last, after the two previous ones.

3.5.7 3DSomaMS: Services required from other Platforms

Image processing and editing.

3.5.8 3DSomaMS: Services provided to other Platforms

- Soma quantitative characterisation.
- Automated typology of somas.
- Simulation of the somas.

3.5.9 3DSomaMS: Prerequisites

3.5.9.1 Required

- R software environment.
- Software for 3D image processing and editing.
- R additional packages.

3.5.9.2 Preferred

None.

3.5.10 3DSomaMS: Necessary Parallel Activities

The 3DSomaMS needs a set of pyramidal neurons in VRML format, provided by SP1.

3.5.11 3DSomaMS: Hardware/Software Functions

This section describes a set of modules, with examples of the key functions they will contain which will be provided by the toolbox.

3.5.11.1 3DSomaMS: Hardware/Software Functions - 3D pyramidal soma representation

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.9	Leader:	Pedro Larranaga
Function Name:	<i>3D pyramidal soma representation</i>		
Test Case:	A user can repair, reconstruct and supervise a soma image for a correct pre-processing (cut of dendrites, filling holes, convex hull and smoothing). He then obtains a representation of the soma as a 3D polyhedron. Jane will receive these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph.		
Planned Start Date:	Mar 2015	Planned Completion Date:	June 2015
Requires Functions:			

3.5.11.2 3DSomaMS: Hardware/Software Functions - Characterisation and typology of pyramidal somas

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.10	Leader:	Pedro Larranaga
Function Name:	<i>Characterisation and typology of pyramidal somas</i>		
Test Case:	A user stores the soma morphological quantitative characteristics. Then he obtains an automatic typology of the soma shapes according to a cluster algorithm and individual somas within each group can be displayed. Jane will receive these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph.		
Planned Start Date:	July 2015	Planned Completion Date:	Oct 2015
Requires Functions:	5.4.2.9		

3.5.11.3 3DSomaMS: Hardware/Software Functions - Pyramidal soma simulator

Task No:	5.4.2	Partner:	UPM-CIG
Function No:	5.4.2.11	Leader:	Pedro Larranaga
Function Name:	<i>Pyramidal soma simulator</i>		
Test Case:	A user has to set the parameters for the optimisation process, which transforms the simulated soma characteristics into a 3D soma image. Jane will receive these probabilities in numerical and graphical formats attached to each node in the directed acyclic graph.		
Planned Start Date:	Nov 2015	Planned Completion Date:	Feb 2016
Requires Functions:	5.4.2.9, 5.4.2.10		



4. Tools for Functional Analysis (WP5.3)

Understanding of brain function depends on data from a wide range of techniques. It is important that simulation results be comparable against these data. To meet this need, the HBP will develop new tools and techniques to compare data from simulations against data from experiments. These will include tools for population analysis (measurement of local field potentials, EEG, fMRI, MEG etc.) (T5.3.1) and tools for the analysis of single cell activity (T5.3.2). Some of these tools will build on previous work in the BrainScaleS project.

4.1 Functional Data Analysis Tools (FDAT): Overall Goals

The NIP will provide a suite of tools, termed the HBP functional data analysis toolbox (FDAT), which makes it possible to analyse and compare multi-scale data on brain dynamics from experiments and from brain simulations. While individual users of the Platform will be able to contribute their own data analysis tools, there is considerable value in having the Platform provide: (a) a set of basic building blocks that can be used to compose more complex analysis workflows, and (b) well-tested implementations of state-of-the-art data analysis methods that offer good performance.

The scope of the functionality to be provided in the first version of the Platform is the analysis of recordings of electrical activity, whether recorded directly using electrodes or multi-electrode arrays, or indirectly through optical imaging of intrinsic or voltage-sensitive dye responses. It is essential that all methods should be applicable to both simulated and experimental data.

More specifically, as specified in the Description of Work, the Platform should provide: tools for the analysis of massively parallel spike train data and local field potentials, including the analysis of population activity and the study of pair- and higher-order correlations; tools for the analysis of LFP (spectral and phase analysis); and tools for processing single unit and patch-clamp recordings.

4.1.1 *FDAT: Use Cases*

4.1.1.1 Full-scale simulation and analysis of a multi-layered local cortical column

Relates to brain simulation subproject, NetSim component, Use Case 1.

Use Case ID:

- SP5FDAT-UC-001

Primary actor:

- Alice, a computational neuroscientist

Description:

- Alice wants to calculate the distributions of spike rates in all populations of neurons (identified by layer and neuron type) in a multi-layered simulation of excitatory and inhibitory neurons. Furthermore, she wants to calculate the average pair-wise cross-correlation histograms between all populations of neurons. Alice wants to perform the analysis using a local script that accesses analysis functions via the REST API of the HBP Platforms.

Preconditions:



- Alice has run a NetSim simulation as described in the corresponding Use Case 1 of the Brain Simulation Subproject, NetSim component. The corresponding simulated spike data are made available by tools developed in other HBP Platforms. Metadata containing cell type, layer and connectivity information are also provided.

Success scenario:

- 1) Alice loads data and metadata.
- 2) Alice constructs a subset of 1000 randomly drawn neurons of each type from each layer based on the provided metadata information.
- 3) For each neuron type and layer:
 - a) Alice selects the corresponding subset of neurons.
 - b) Alice uses a rate estimator of the HBP functional data analysis toolbox to calculate the distribution of firing rates of that set of neurons.
 - c) Alice calculates the cross-correlation histogram between each pair of spike trains in the set of neurons and averages the result.
 - d) The distribution of firing rates and the distribution of cross-correlation histograms and its average are saved to a file together with information regarding the neuron type and layer used for their calculation.
- 4) The final result file is made available for further analysis using the tools developed in other HBP Platforms.

Post-conditions:

- Firing rate distributions and cross-correlations are available via tools of the HBP Platforms. They can be visualised using standard plotting functions on the portal website.

Alternate scenarios:

- Data are too large to fit in memory
 - The system produces an error message, suggesting the use of a different workflow, involving processing the data in chunks.
- Data are not saved to file, but stored in memory for further processing.
 - Skip step (3d) and (4).
- Computation time is too long.
 - Alice should redesign the analysis in step (3) such that firing rates and cross-correlations are calculated for one layer and one neuron population only (as specified by the user).
 - Alice may then separately and independently execute the redesigned analysis on multiple machines, specifying for each execution a different combination of layer and population.

4.1.1.2 Web-based analysis of simulated data**Use Case ID:**

- SP5FDAT-UC-002

Primary actor:

- Christina, a theoretical neuroscientist

**Description:**

- In order to test a theoretical prediction, Christina wants to determine whether in a given simulation experiment action potentials are associated with an increase in the excitatory synaptic conductance or a decrease in the inhibitory conductance.

Preconditions:

- Results from a detailed simulation are available in an HDF5 file in the HBP DataSpace. For each cell type in the model, the membrane potentials of ten neurons were recorded, together with the post-synaptic conductances of each synapse onto that neuron. For each trace, the HDF5 file contains information about the cell and synapse type with which it is associated.

Success scenario:

- 1) Using the HBP Collaboratory, Christina:
 - a) Loads the data and metadata from file. For each synapse recording, she extracts the conductance trace for the 10 msec preceding each spike, and averages over all spikes using a function of the HBP functional data analysis toolbox. These data are written to a new HDF5 file in the DataSpace, together with the associated metadata propagated from the original file.
 - b) Loads the data from the previous step and averages over all synapses of a given type on each neuron. This data, with metadata, are again written to the DataSpace.
 - c) Loads the data from the previous step and averages over all neurons of a given type. The resultant data—one average conductance trace per synapse type-neuron type pair—are written to an HDF5 file in the DataSpace.
- 2) Since they have a recognised semantic data type (time series), the results can be automatically visualised through the Portal.

Alternate scenarios:

- The time window to cut around each spike is chosen too large, such that there is no data for some values of the lag.
 - The spike-triggered average function ignores spikes where the problem arises and produces a warning message.

4.1.1.3 Estimating complex stimulus-response relationships**Use Case ID:**

- SP5FDAT-UC-003

Primary actor:

- Zachary, a computational neuroscientist working on models of the visual system

Description:

- Zachary wants to produce an orientation preference map for the model, so that he can compare it to experimental data. Since this computation will be performed many times, he wishes to encapsulate the entire analysis as a single function and add it to the portal.

Preconditions:

- Zachary has run a simulation in which the model retina is presented with a series of full-field drifting gratings, with eight different orientations. Between each grating



presentation, the retina sees a blank screen of background luminance for one second. Spikes have been recorded for all neurons in layer 2/3 of primary visual cortex, and saved to file in HDF5 format. The HDF5 file also contains the (x, y) position of each neuron and the onset time, duration and orientation of each grating presentation.

Success scenario:

- 1) Zachary writes a short Python script that uses a number of functions from the NIP functional data analysis toolbox. The script does the following:
 - a) Loads the spike train data and metadata.
 - b) For each neuron:
 - Calculates the average firing rate during each stimulus presentation, subtracting the firing rate during the blank screen, to produce a curve of response amplitude vs. orientation (tuning curve).
 - Fits the von Mises distribution to the tuning curve and takes the orientation value δ at the peak of the distribution.
 - Creates a table of (x, y, δ) for all neurons, and writes it to a text file.
- 2) Zachary tests the script locally then uses the Task registration tool of the Collaboratory to publish his tool to the HBP Platforms Task Repository.
- 3) He then uses the tool to calculate the (x, y, δ) table, downloads it to his local computer, and uses his preferred plotting tool to plot the data as an orientation preference map.

Alternate scenarios:

- The calculation is too slow.
 - Zachary parallelises the script: each neuron can be processed independently. By using the HDF5 format, parallel IO is also possible.

4.1.1.4 Simulator development**Use Case ID:**

- SP5FDAT-UC-004

Primary actor:

- Bernadette, a simulator developer

Description:

- Bernadette wants to calculate the histogram of the coefficient of variation (CV) of the inter-spike interval to check that the simulator behaviour has not changed after a refactoring. Since this test will be repeated many times, she wishes to automate it.

Preconditions:

- Bernadette has run a simulation of a balanced random network with N=10000 neurons, and recorded spikes from 10% of the neurons. The spike times are saved in an HDF5 file in the HBP DataSpace.

Success scenario:



- 1) Bernadette writes a script, using the HBP REST API, which does the following:
 - a) Bernadette writes a script, using the HBP Portal Rest API, which does the following:
 - b) Loads the spike train data from an HDF5 file.
 - c) For each spike train:
 - Calculates the inter-spike intervals.
 - Calculates the CV for each set of intervals.
 - Constructs the histogram of the cvs in the complete population.
 - Saves the histogram to a text file in the HBP dataspace.

2) The script returns the URL of the output data file.

Post-conditions:

- Bernadette can compare the contents of the returned text file containing the histogram data to previously computed versions of that file.

Alternate scenarios:

- 1) The HDF5 file is incorrectly formatted.
 - a) The script halts with an informative error message.
- 2) Some of the neurons were silent, and so produced spike trains of length zero.
 - a) For each empty spike train:
 - The inter-spike interval function returns an empty array and produces a warning message.
 - The coefficient of variation function returns NaN and produces a warning message.
 - The histogram function ignores the NaN values and produces a warning message.
- 3) Due to a bug introduced during the refactoring, all of the neurons were silent.
 - a) Since all the values given to the histogram function are NaN, the output data file is empty. The log file can be used to understand why this has happened.

4.1.1.5 LFP analysis using data from an electrophysiological experiment

Use Case ID:

- SP5FDAT-UC-005

Primary actor:

- Eric, an electrophysiologist

Description:

- Eric wants to calculate the time-resolved LFP power in experimentally recorded electrophysiological data, and contrast differences in power between two behavioural conditions (A and B) during the experiment.

Preconditions:



- Eric has performed the experiment. The corresponding continuously recorded LFP data are available on the HBP DataSpace.
- Metadata containing the definition of a behavioural trial in terms of the experimental trigger events, the exact behavioural condition of each trial, and the position of recording electrodes is also available on the via the HBP DataSpace.

Success scenario:

- 1) Using the Collaboratory web interface, Eric:
 - a) Loads the data and metadata.
 - b) Uses the HBP functional data analysis toolbox to cut the data into individual trials. Trials are retained in a data structure provided by the toolbox.
 - c) Selects the two subsets of trials corresponding to behavioural conditions a and b, respectively, based on the behavioural meta-information attached to each trial.
 - d) Calculates the average power spectrum of the trials separately for trials belonging to condition a) and b) using functions of the nip functional data analysis toolbox.
 - e) Saves the two average power spectra to the HBP dataspace, including metadata identifying the recording session, and differentiating conditions A and B.

Post-conditions:

- Average power spectra of the LFPs for the two sets of trials corresponding to conditions A and B, annotated with metadata, are available in a file and accessible through the HBP portal.

Alternate scenarios:

- 1) Data are too large to fit in memory.
 - a) The system produces an error message, suggesting use of a different workflow involving processing the data in chunks.
- 2) Eric uses different (behavioural) meta-information than attached to each trial.
 - a) Error message by the system, or the system returns a data structure that is sufficient for both for Eric and the system to work on further.
- 3) Sampling frequencies of the LFPs are different.
 - a) Early error message (before cutting to trials) to adjust the frequency.
 - b) Down sampling to the lowest sampling rate of the signals.
- 4) Data are not saved to file, but stored in memory for further processing.
 - a) Skip step (1e).



4.1.1.6 Spike sorting

Use Case ID:

- SP5FDAT-UC-006

Primary actor:

- Liang, an experimentalist recording from rat somatosensory cortex

Description:

- Liang, an experimentalist recording from rat somatosensory cortex using Utah arrays.

Preconditions:

- Liang has a data file in Blackrock format containing data from twenty trials, which he has uploaded to the HBP DataSpace.

Success scenario:

- 1) Using the HBP Collaboratory, Liang runs an automated spike-sorting tool using the default parameters.
- 2) For each putative neuron detected in the recording, the tool produces a spike train data object containing the spike times and the waveform of each spike.
- 3) The tool saves this data to an HDF5 file in the HBP DataSpace together with metadata indicating from which electrode each spike train was constructed.
- 4) The tool also generates two figures in PNG format. One plots the spikes' projections in the feature space (where the features may include peak-to-peak amplitude and projections on the principal components). The second plots the aligned and overlapped spike waveforms.
- 5) Based on these figures, Liang determines that the spike sorting could be improved. Using the Collaboratory he adjusts some of the parameters and re-runs the analysis until he is happy with the results.

Post-conditions:

- Liang has access to an HDF5 file in the HBP DataSpace, containing spike times, spike waveforms and metadata.

Alternate scenarios:

- 1) Liang cannot obtain a satisfactory result using the Portal, and wishes to work more interactively.
 - a) He downloads and installs the HBP functional data analysis toolbox on his own computer, which gives a shorter feedback loop and enables him to inspect the data at a finer scale.
 - b) When he is happy with the results, he re-runs the analysis using the Portal, using the parameters he has determined interactively, so as to take advantage of the Portal's provenance tracking capabilities.

4.1.1.7 Detecting higher-order spike patterns in stochastic model data

Use Case ID:

- SP5FDAT-UC-007

Primary Actor:



- Borislav, a computational neuroscientist interested in learning more about methods to detect higher-order spike patterns in data

Description:

- Borislav wants to investigate the ability of the frequent itemset mining based spike pattern analysis to detect the repeated synchronous firing of a subset of K neurons in his simulated data of M neurons (homogeneous firing rates and stationary in time).

Preconditions:

- Borislav has extracted the relevant parameters from the simulation (simulation time t , average firing rate R) and stored them in a local HDF5 file.

Success scenario:

- 1) Borislav reads t and R from file.
 - a) Borislav writes and locally executes a script in which he performs the following steps $h=1000$ times for each of the two free parameters r (rate of fully synchronous spike events) and $K=1..M$:
 - b) Perform a stochastic simulation of length t of the network activity using functions supplied by the toolbox, where:
 - $M-K$ neurons are simulated as a Poisson process of rate R .
 - K neurons are modelled as a Poisson process with rate $R-r$, with a fully synchronous spike pattern injected into all K neurons according to a Poisson process at rate r (i.e., a single interaction process with background spiking).
 - c) Run the frequent itemset spike pattern detection on the data.
 - d) Evaluate the significance using surrogates of all found patterns.
 - e) Determine and save if the injected spike pattern of the K neurons is classified as significant.
- 2) For each tuple (r,K) the script then determines the percentage of simulations where the injected pattern was detected as significant.
- 3) The resulting matrix is saved to an HDF5 file on disk.

Post-condition:

- Borislav has access to the result of the analysis located in a local file for further analysis or plotting.

4.1.2 FDAT: Functional Requirements

4.1.2.1 Overview

The HBP functional data analysis toolbox shall provide a set of functions for analysing recordings of electrical activity obtained from experiments or simulations.

4.1.2.2 Acceptable data types

Functional Requirement ID:

- SP5FDAT-FR-001

Features:

- Inputs to and outputs from analysis functions shall be of one of the following types:



- Integers.
 - Floating point numbers (up to 64-bit precision).
 - Complex numbers.
 - Unicode strings.
 - Dates/times in iso 8601 format.
 - N-d arrays containing any of the above data types.
 - Physical units (the representation of units is specified below).
 - Tables in which each column has a well-defined data type.
 - Spike trains (the representation of a spike train is specified below).
 - Analogue signals (the representation of an analogue signal is specified below).
 - Events (the representation of an event is specified below).
 - Lists/tuples containing any of the above data types.
 - Dictionaries/hash-tables containing any of the above data types.
- Physical units shall use the SI system.
 - Physical units shall be represented as unicode strings. The letter “u” may be used in place of the prefix “μ” (micro).
 - Analogue signals shall be compound data types representing one or more continuous, analogue signals (note that “continuous” and “analogue” refer to the underlying physical signal, not to its representation, which will of course be quantised and sampled. “Time series” is an alternative name). Analogue signals must contain physical units for the time and value dimensions, and either a sampling rate, a sampling interval, or an explicit array of sample times.
 - Spike trains shall be compound data types containing at minimum a float array of spike times and the physical units for the time dimension. A spike train may also contain a waveform for each spike, represented as an analogue signal.
 - Events shall be compound data types containing a time quantity (floating point number plus unit) and a unicode label.

4.1.2.3 Necessary metadata

Functional Requirement ID:

- SP5FDAT-FR-002

Features:

- The toolbox shall require that all data files provide the minimal metadata specified for in the HBP-CORE *minimum metadata specification*.



4.1.2.4 File formats

Functional Requirement ID:

- SP5FDAT-FR-003

Features

- The toolbox shall provide functions to read electrophysiology data and associated metadata from any of the following file formats:
 - Blackrock Microsystems format (.nev/.nsX).
 - Elphy .dat format.
 - HDF5 structured according to the NeoHdf5IO specification.
 - <http://neo.readthedocs.org/en/latest/io.html#neo.io.NeoHdf5IO>
 - HDF5 structured according to the recommendations of the INCF Task Force on Electrophysiology Data Sharing.
 - Matlab .mat files structured according to the NeoMatlabIO specification.
 - <http://neo.readthedocs.org/en/latest/io.html#neo.io.NeoMatlabIO>
- The toolbox shall provide functions to write spike train, analogue signal and event data, together with associated metadata, to any of the following file formats:
 - HDF5 structured according to the NeoHDF5IO specification.
 - HDF5 structured according to the recommendations of the INCF Task Force on Electrophysiology Data Sharing.
 - Matlab .mat files structured according to the NeoMatlabIO specification.
- The toolbox shall provide functions to read tabular or array data from any of the following file formats:
 - HDF5
 - Comma-separated/tab-separated format using unicode encoding.
- The toolbox shall provide functions to write tabular or array data to any of the following file formats:
 - HDF5
 - Comma-separated/tab-separated format using unicode encoding.
- We recommend the use of HDF5 serialisation as a format for the efficient storage and exchange of data objects across tools developed by HBP Platforms.
- We recommend that metadata should be provided in an odML format.

4.1.3 *FDAT: Non-Functional Requirements*

4.1.3.1 Interfaces

- All functions shall be available for use through the HBP Collaboratory (both Web GUI and programmatic web service clients).
- All functions shall also be available for use by downloading and installing a Python library on a local computer.



4.1.3.2 Efficiency

Where execution of a function could take more than 10 seconds (as measured on a typical laptop computer) and an embarrassingly parallel parallelisation approach is feasible, the function shall be parallelised using the methods available in the Collaboratory SDK.

4.1.3.3 Reliability/correctness

- All functions shall be tested with at least one typical input, two edge-case inputs, and two invalid inputs.
- All branch points in all functions shall be tested.

4.1.3.4 Documentation

- All functions shall be clearly documented, both in the source code and in web-based documentation.
- Further documentation shall explain how functions may be combined in simple workflows.

4.1.3.5 Debugging and provenance tracking

- Analysis functions shall produce log files to assist in debugging.
- Executing an analysis function in the Portal environment will provide a provenance trail by making use of the Portal SDK.
- All analysis functions shall be verified by unit tests.

4.1.4 *FDAT: Software*

The principal architectural constraint is the need for analysis functions to be available both as Tasks in the Collaboratory and as standalone functions for local use. The toolbox will therefore be implemented as three separate Python packages. The first package, henceforth called the Base package, will provide a platform-agnostic implementation of each analysis function, except for spike sorting. The second package, called the Spike Sorting package, will extend the Base package with functionality specific to spike sorting. The third package, called the Interface package, will wrap all public functions from the Base and Spike Sorting packages so as to make them available as Collaboratory Tasks.

To monitor the progress of work, we define three Key Performance Indicators (KPIs) for the HBP Functional Data Analysis Toolbox.

- 1) The number of completed functions listed under FDAT: Hardware/Software Functions.
- 2) The number of completed (python) functions in the source code.
- 3) The total test coverage.

4.1.4.1 Base package

The base package:

- Will be based on the standard scientific Python stack (numpy, scipy) etc. And on the Neo electrophysiology data-handling package;



- Will be developed as open-source software in order both to maximise the impact of this HBP activity on the wider community and to benefit from external expertise and testing;
- Will support Python versions 2.7, 3.3 and any newer Python versions as they are released;
- Will be tested on Linux, OS X and Windows;
- Will be documented using Sphinx;
- Will use Git for version control.

4.1.4.2 Spike sorting package

For the spike sorting package:

- We will either use an existing open source Python spike sorting package (e.g. Spikesort - <http://spikesort.org>), or we will develop a wrapper package that integrates functionality from more than one existing package;
- We will work with the authors of existing packages to add any additional functionality required (e.g. Support for Neo objects);
- Python version and OS support, documentation and version control will be as for the base package.

4.1.4.3 Interface package

This package:

- Will wrap all public functions from the Base and Spike Sorting packages so as to make them available as Collaboratory Tasks;
- Will support Python version 2.7;
- Will be tested on Linux;
- Will use the documentation mechanisms of the Collaboratory;
- Will use Git for version control.

4.1.5 *FDAT: Physical Architecture*

FDAT will support Python versions 2.7, 3.3 (and any newer Python versions as they are released) on standard 32bit/64bit desktop or server hardware running on Linux, OS X and Windows.

4.1.6 *FDAT: Services required from other components of the NIP*

- Data storage - Shared Data Space component.
- Data standards.

4.1.7 *FDAT: Services required from other Platforms*

- Provenance tracking - Collaboratory.
- Task organisation, tracking and launching interfaces - Collaboratory.

- Web based data visualisation services - Collaboratory.
- Job execution - HPC.

4.1.8 *FDAT: Services provided to other Platforms*

Collaboratory - functional data analysis tools.

4.1.9 *FDAT: Prerequisites*

FDAT requires the specifications of platform tools to integrate the toolbox functions into the Portal.

4.1.10 *FDAT: Necessary Parallel Activities*

FDAT needs to be used and evaluated by data analysts during the development process in order to continuously adapt design decisions during development according to user feedback. To this end, both recorded and simulated multiple channel spike and LFP data for analysis need to be provided to the HBP Platforms.

4.1.11 *FDAT: Hardware/Software Functions*

This section describes a set of modules, with examples of the key functions they will contain which will be provided by the toolbox.

4.1.11.1 *FDAT: Hardware/Software Functions - Signal processing*

This module shall provide functions to manipulate time series represented as analogue signals. Key functionalities:

- Calculating the z-score of a given signal.
- Frequency filtering of signals (low pass, high pass, band pass and band stop filters).
- Calculating the analytic signal of a signal, which yields phase and amplitude information.

Task No:	5.3.1	Partner:	CNRS, Jülich
Function No:	5.3.1.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Signal processing (basic)		
Test Case A:	A user can calculate the z-score of a given signal.		
Test Case B:	A user can filter the frequency of signals.		
Planned Start Date:	July 2014	Planned Completion Date:	October 2014
Requires Functions:			

Task No:	5.3.1	Partner:	CNRS, Jülich
Function No:	5.3.1.2	Leader:	Andrew Davison, Sonja Grün

Function Name:	Signal processing (advanced)		
Test Case:	A user can calculate the analytic signal (phase and amplitude).		
Planned Start Date:	November 2014	Planned Completion Date:	August 2015
Requires Functions:	Function 5.3.1.1.		

Task No:	5.3.1	Partner:	CNRS, Jülich
Function No:	5.3.1.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Signal processing (integrated)		
Test Case:	All functions of 5.3.1 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	August 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.1.1. & 5.3.1.2		

4.1.11.2 FDAT: Hardware/Software Functions - Frequency and power-spectrum analysis

This module shall provide functions for the spectral analysis of time series represented as analogue signals. Key functionalities:

- Calculation of the Fourier transform and power spectrum of a signal.
- Calculation of the coherence and coherency between two signals.

Task No:	5.3.2	Partner:	CNRS, Jülich
Function No:	5.3.2.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Frequency and power-spectrum analysis (basic)		
Test Case A:	A user can calculate the Fourier transform of a signal.		
Test Case B:	A user can calculate the power spectrum of a signal.		
Planned Start Date:	October 2014	Planned Completion Date:	April 2015
Requires Functions:			

Task No:	5.3.2	Partner:	CNRS, Jülich
Function No:	5.3.2.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Frequency and power-spectrum analysis (advanced)		
Test Case A:	A user can calculate the coherence between two signals.		
Test Case B:	A user can calculate the coherency between two signals.		
Planned Start Date:	April 2015	Planned Completion Date:	October 2015
Requires Functions:	Function 5.3.2.1		

Task No:	5.3.2	Partner:	CNRS, Jülich
Function No:	5.3.2.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Frequency and power-spectrum analysis (integrated)		
Test Case:	All functions of 5.3.2 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	October 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.2.1 & 5.3.2.2		

4.1.11.3 FDAT: Hardware/Software Functions - Spike train statistics

This module shall provide measures that describe statistical properties of single or multiple parallel spike trains. Key functionalities:

- Calculation of the time series of inter-spike intervals.
- Calculation of the coefficient of variation and the Fano factor.
- Spike train metrics, such as the Victor-Purpura or van Rossum spike distances.

Task No:	5.3.3	Partner:	CNRS, Jülich
Function No:	5.3.3.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train statistics (basic)		
Test Case A:	A user can calculate the time series of inter-spike intervals.		
Test Case B:	A user can calculate the coefficient of variation and the Fano factor.		
Planned Start Date:	April 2014	Planned Completion Date:	July 2014
Requires Functions:			

Task No:	5.3.3	Partner:	CNRS, Jülich
Function No:	5.3.3.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train statistics (advanced)		
Test Case:	User can calculate train metrics e.g., Victor-Purpura or van Rossum spike distances.		
Planned Start Date:	July 2014	Planned Completion Date:	April 2015
Requires Functions:	Function 5.3.3.1		

Task No:	5.3.3	Partner:	CNRS, Jülich
Function No:	5.3.3.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train statistics (integrated)		
Use Case A:	All functions of 5.3.3 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	April 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.3.1 & 5.3.3.2		

4.1.11.4 FDAT: Hardware/Software Functions - Rate estimation

This module provides different functions to compute the instantaneous firing rate of a single spike train or the population rate and complexity distributions of multiple parallel spike trains. This module also provides functions to calculate the peri-stimulus time histogram (PSTH), which measures the trial-averaged response rate response to a trigger. Algorithms used for rate estimation:

- Discrete temporal binning of spike times.
- Convolution of spikes with smooth kernels, such as Gaussian windows, to create a continuous rate estimate.

Task No:	5.3.4	Partner:	CNRS, Jülich
Function No:	5.3.4.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Rate estimation (basic)		
Test Case:	A user can calculate the discretely binned representation of spike trains.		
Planned Start Date:	April 2014	Planned Completion Date:	July 2014
Requires Functions:			

Task No:	5.3.4	Partner:	CNRS, Jülich
Function No:	5.3.4.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Rate estimation (advanced)		
Test Case:	A user can create a continuous rate estimate, using convolution of spikes with smooth kernels, such as Gaussian windows.		
Planned Start Date:	July 2015	Planned Completion Date:	April 2015
Requires Functions:	Function 5.3.4.1		

Task No:	5.3.4	Partner:	CNRS, Jülich
Function No:	5.3.4.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Rate estimation (integrated)		
Test Case:	All functions of 5.3.4 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	April 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.4.1 & 5.3.4.2		

4.1.11.5 FDAT: Hardware/Software Functions - Spike train correlation analysis

This module shall provide functions to compute measures of correlation between spike trains:

- Covariance between two parallel spike trains.
- Cross-correlation histogram between two parallel spike trains.
- Matrices of covariances or correlation coefficients between multiple parallel spike trains.

Task No:	5.3.5	Partner:	CNRS, Jülich
Function No:	5.3.5.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train correlation analysis (basic)		
Test Case A:	A user can calculate the covariance between two parallel spike trains.		
Test Case B:	A user can calculate the cross-correlation histogram between two parallel spike trains.		
Planned Start Date:	October 2014	Planned Completion Date:	January 2015
Requires Functions:			

Task No:	5.3.5	Partner:	CNRS, Jülich
Function No:	5.3.5.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train correlation analysis (advanced)		
Test Case A:	A user can calculate matrices of covariances between multiple parallel spike trains.		
Test Case B:	A user can calculate matrices of correlation coefficients between multiple parallel spike trains.		
Planned Start Date:	January 2015	Planned Completion Date:	October 2015
Requires Functions:	Function 5.3.5.1		

Task No:	5.3.5	Partner:	CNRS, Jülich
Function No:	5.3.5.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike train correlation analysis (integrated)		
Test Case:	All functions of 5.3.5 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	October 2016	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.5.1 & 5.3.5.2		

4.1.11.6 FDAT: Hardware/Software Functions - Spike-triggered averaging

This module shall provide functions to calculate the correlation between a point process (e.g., a spike train) and a time series represented as an analogue signal (e.g., an LFP). Thus it provides common diverse measures such as the spike-triggered average LFP, the spike-triggered stimulus intensity, or the event-triggered LFP. Key functionalities:

- Spike-triggered average
- Spike-field coherence

Task No:	5.3.6	Partner:	CNRS, Jülich
Function No:	5.3.6.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike triggered averaging (basic)		
Test Case:	A user can calculate the spike-triggered average.		
Planned Start Date:	July 2014	Planned Completion Date:	October 2014
Requires Functions:			

Task No:	5.3.6	Partner:	CNRS, Jülich
Function No:	5.3.6.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike triggered averaging (advanced)		
Test Case:	A user can calculate the spike-field coherence.		
Planned Start Date:	October 2014	Planned Completion Date:	July 2015
Requires Functions:	Function 5.3.6.1		

Task No:	5.3.6	Partner:	CNRS, Jülich
Function No:	5.3.6.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Spike triggered averaging (integrated)		
Test Case:	All functions of 5.3.6 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	July 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.6.1 & 5.3.6.2		

4.1.11.7 FDAT: Hardware/Software Functions - Spike pattern analysis using frequent itemset mining (FIM)

This module shall provide a tool to extract patterns of synchronously active neurons from massively parallel spike trains, and to evaluate their significance based on the number of pattern occurrences and pattern complexity using surrogate data. Key functionalities:

- Extract sets of synchronously active neurons and occurrence counts.
- Evaluate significance of observed spike patterns.

Task No:	5.3.7	Partner:	Jülich
Function No:	5.3.7.1	Leader:	Sonja Grün
Function Name:	Frequent itemset mining (basic)		
Test Case:	A user can detect frequent patterns in synchronously active neurons.		
Planned Start Date:	October 2014	Planned Completion Date:	April 2015
Requires Functions:			

Task No:	5.3.7	Partner:	Jülich
Function No:	5.3.7.2	Leader:	Sonja Grün
Function Name:	Frequent itemset mining (advanced)		
Test Case:	A user can evaluate significance of observed spike patterns		
Planned Start Date:	April 2015	Planned Completion Date:	October 2015
Requires Functions:	Function 5.3.7.1		

Task No:	5.3.7	Partner:	Jülich
Function No:	5.3.7.3	Leader:	Sonja Grün
Function Name:	Frequent itemset mining (integrated)		
Test Case:	All functions of 5.3.7 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	October 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.7.1 & 5.3.7.2		

4.1.11.8 FDAT: Hardware/Software Functions - Stochastic process generation

This module shall provide functionality to generate simulated spike trains and simulated multiple parallel spike trains according to statistical models. Key functionalities include:

- Generation of one or more independent Poisson/gamma distributed time series.
- Functions to generate correlated sets of (multiple) multidimensional Poisson processes according to the SIP (single interaction process) and MIP (multiple interaction process) models.

Task No:	5.3.8	Partner:	CNRS, Jülich
Function No:	5.3.8.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Stochastic process generation (basic)		
Test Case A:	A user can generate one or more independent Poisson distributed time series.		
Test Case B:	A user can generate one or more independent Gamma distributed time series.		
Planned Start Date:	October 2014	Planned Completion Date:	January 2015
Requires Functions:			

Task No:	5.3.8	Partner:	CNRS, Jülich
Function No:	5.3.8.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Stochastic process generation (advanced)		
Test Case A:	A user can generate correlated sets of (multiple) multidimensional Poisson processes according to the SIP (single interaction process) model.		
Test Case B:	A user can generate correlated sets of (multiple) multidimensional Poisson processes according to the MIP (multiple interaction process) model.		
Planned Start Date:	January 2015	Planned Completion Date:	October 2015
Requires Functions:	Function 5.3.8.1		

Task No:	5.3.8	Partner:	CNRS, Jülich
Function No:	5.3.8.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Stochastic process generation (integrated)		
Test Case:	All functions of 5.3.8 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	October 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.8.1 & 5.3.8.2		

4.1.11.9 FDAT: Hardware/Software Functions - Surrogate generation

This module shall provide functions to generate surrogates of spike trains that can be used by other analysis functions to generate distributions for hypothesis testing. Key surrogates to implement:

- Surrogates obtained by uniformly dithering times around the original position.
- Surrogates obtained by randomly shuffling spike times within the spike train.
- Surrogates obtained by uniformly shifting each trial by a random time offset.

Task No:	5.3.9	Partner:	CNRS, Jülich
Function No:	5.3.9.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Surrogate generation (basic)		
Test Case A:	A user can obtain surrogates by uniformly dithering times around the original position.		
Test Case B:	A user can obtain surrogates by randomly shuffling times around the original position.		
Planned Start Date:	October 2014	Planned Completion Date:	April 2015
Requires Functions:			



Task No:	5.3.9	Partner:	CNRS, Jülich
Function No:	5.3.9.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Surrogate generation (advanced)		
Test Case:	A user can obtain surrogates by uniformly shifting each trial by a random time offset.		
Planned Start Date:	April 2015	Planned Completion Date:	January 2016
Requires Functions:	Function 5.3.9.1		

Task No:	5.3.9	Partner:	CNRS, Jülich
Function No:	5.3.9.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Surrogate generation (integrated)		
Test Case:	All functions of 5.3.9 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	January 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.9.1 & 5.3.9.2		

4.1.11.10 FDAT: Hardware/Software Functions Utility functions

This module shall contain data-manipulation and low-level, non-neuroscience-specific analysis functions. Key functionality:

- Creating a histogram of values.
- Calculating statistical measures such as the mean, median, standard deviation, variance and standard error of the mean.
- Selecting subsets of data (such as specific electrodes, specific neurons, specific time windows).

Task No:	5.3.10	Partner:	CNRS, Jülich
Function No:	5.3.10.1	Leader:	Andrew Davison, Sonja Grün
Function Name:	Utility functions (basic)		
Test Case A:	A user can create a histogram of values.		
Test Case B:	A user can calculate statistical measures such as mean, standard deviation, variance and standard error of mean.		
Planned Start Date:	April 2014	Planned Completion Date:	January 2015
Requires Functions:			

Task No:	5.3.10	Partner:	CNRS, Jülich
Function No:	5.3.10.2	Leader:	Andrew Davison, Sonja Grün
Function Name:	Utility functions (advanced)		
Test Case:	A user can select subsets of data (such as specific electrodes, specific neurons and specific time windows).		
Planned Start Date:	January 2015	Planned Completion Date:	October 2015
Requires Functions:	Function 5.3.10.1		

Task No:	5.3.10	Partner:	CNRS, Jülich
Function No:	5.3.10.3	Leader:	Andrew Davison, Sonja Grün
Function Name:	Utility functions (integrated)		
Test Case:	All functions of 5.3.10 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	October 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.10.1 & 5.3.10.2		

4.1.11.11 FDAT: Hardware/Software Functions Spike sorting

This module shall provide an interface to functions that perform spike sorting of data consisting of neuronal spike waveforms and spike time points.

Task No:	5.3.11	Partner:	CNRS
Function No:	5.3.11.1	Leader:	Andrew Davison
Function Name:	Spike sorting		
Test Case:	A user can apply a spike-sorting algorithm based on Neo compatible data and obtains results in a Neo data structure.		
Planned Start Date:	July 2014	Planned Completion Date:	July 2015
Requires Functions:			

Task No:	5.3.11	Partner:	CNRS
Function No:	5.3.11.2	Leader:	Andrew Davison
Function Name:	Spike sorting (integrated)		
Test Case:	All functions of 5.3.11 are fully tested and integrated into the HBP Platforms.		
Planned Start Date:	July 2015	Planned Completion Date:	April 2016
Requires Functions:	Function 5.3.11.1		



5. Predictive Neuroinformatics (WP5.4)

The HBP will make a major effort to develop new tools for Predictive Neuroinformatics, using machine learning and statistical modelling techniques to extract rules describing the relationships between data sets for different levels of brain organisation. A statistical model will make it possible to derive a “global neuronal addressing system” that can predict the targets and the course of long-range axonal projections from specific types of neurons to their targets in the rest of the brain (T5.4.1). A second important goal will be to develop algorithms that use the statistical structure of neuronal and synaptic geometries to computationally synthesise model neurons and synapses (T5.4.2).

5.1 Predictive Neuroscience: Brain Addressing System

The Brain Addressing System builds connection matrices encapsulated in the pathway data object for use in the simulation and provides tools for validation of the instantiated connectivity.

5.1.1 Brain Addressing System: Overall Goals

The Brain Addressing System is intended to produce the synaptic connection matrix between two populations of neurons, defined in terms of source and destination area and source and destination cell type. It will specify between which presynaptic and postsynaptic branch the synaptic connection is made. The Brain Addressing System is an implementation of a statistical model that incorporates available experimental data.

5.1.2 Brain Addressing System: Use Cases

5.1.2.1 Building the projection from thalamocortical relay cells in the ventral posterior medial nucleus of the thalamus to layer 4 spiny stellate cells in the barrel cortex

Use Case ID:

- SP5BAS-UC-001

Primary Actor:

- One scientific user

Success Scenario:

- Scientific developer user Daniel would like to generate an instantiation of the projection of thalamocortical relay cells from the ventral posterior medial (VPM) nucleus of the thalamus to layer 4 spiny stellate cells (L4 SS) in barrel cortex (S1) in the rodent.
- Daniel will select the source area (VPM) and cell type (relay cell) and the destination area (S1) and cell type (L4 SS) in the atlasing tool and download a *projection* data object with the corresponding statistics: size of area, cell density, axonal/dendritic morphology.
- Daniel will call the connectivity engine to build connectivity based on the downloaded statistics, which will generate indices for each cell, morphologies, and a connection matrix. For each instantiated connection, it will list the axonal branch on which the presynaptic part of the synapse is located, and the dendritic branch on which the



presynaptic part is located. This information will be collected in a *pathway* data object.

- Daniel wants to validate the connectivity. He uploads the data object to the *pathway validator* tool—which calculates relevant statistics such as distribution of synapses across branch order—which he compares to experimental data in the atlas. The outcome of the validation is placed in a *pathway validation* data object.
- Finally, he incorporates the *pathway* data object into the circuit building workflow to include the synthesis task in the appropriate location.

5.1.2.2 Validating the thalamocortical projection generated by the brain builder

Use Case ID:

- SP5BAS-UC-002

Success Scenario:

- Scientific Developer User Jezebel has build a mouse brain using the brain builder and would like to validate the thalamocortical pathway in her model.
- Jezebel will select the source area (VPM) and cell type (relay cell) and the destination area (S1) and cell type (L4 SS) in the atlasing tool and download a *projection* data object with the corresponding statistics: the branch order statistics of the projection.
- Jezebel collects the relevant part of her connection matrix into a *pathway* data object
- She then uploads the *pathway* data object to the *pathway validator* tool, which returns the *pathway validation* data object.
- She inspects the *pathway validation* data object using the visualisation tool and concludes that the instantiated projection does not match the recently updated experimental data.

5.1.3 Brain Addressing System: Functional Requirements

5.1.3.1 Authentication and authorisation

Functional Requirement ID:

- SP5BAS-FR-001

Features:

- Relying on the SSO Authentication and Authorisation functionality of the Collaboratory.
- Satisfies all Functional Requirements for Authorisation and Authentication of the Collaboratory.

5.1.3.2 Selection of projection

Functional Requirement ID:

- SP5BAS-FR-002

Features:



- The source area and cell type, and destination area and cell type can be selected in the Atlasing tool using GUI or scripting. The naming conforms to the brain region ontology from the KnowledgeSpace.
- The Atlasing tool will return a Projection data object for the selected areas. The projection data object will represent data from various sources such as the Allen projectome data set.
- A projection data object inspection tool will need to be provided for a check of the data object.

5.1.3.3 Generation of projection

Functional Requirement ID:

- SP5BAS-FR-002

Features:

The BrainAddressingSystem will accept a *projection* data object and an empty or incomplete *pathway* data object and return a completed *pathway* data object.

- 1) The *projection* data object (input) will need to be inspected to determine if it is completely specified for further processing; the input *pathway* data object will need to be checked for consistency with the specified sending and receiving areas.
- 2) The appropriate number of sending and receiving cells will need to be created together with an index for each, if not already provided in the input *pathway* data object.
- 3) The corresponding number of presynaptic elements (boutons) and postsynaptic elements (spines for excitatory synapses) will need to be calculated.
- 4) For each neuron, the voxel-based distribution of presynaptic or postsynaptic elements will need to be generated.
- 5) For each neuron, the appropriate number of presynaptic or postsynaptic elements will need to be generated based on the voxelised distribution.
- 6) Each presynaptic element will need to be paired to a postsynaptic element
- 7) The axonal or morphology will need to be generated based on the position of the synapses assigned in step 6 and those previously assigned. This will assign a branch index to each presynaptic and postsynaptic element.

5.1.3.4 Validation of projection

Functional Requirement ID:

- SP5BAS-FR-004

Features:

The BrainAddressingSystem will accept a *projection* data object and *pathway* data object and return a *pathway validation* data object.

- 1) The *projection* and *pathway* data object will need to be inspected to determine if it is completely specified for further processing
- 2) Statistics on synapse placement will need to be determined based on the *pathway* data object and compared to experimental data extracted from the *projection* data object.
- 3) The degree of correspondence between instantiated connectivity and experimental data will need to be quantified and summarised in the *pathway validation* data object.



5.1.4 Brain Addressing System: Non-Functional Requirements

5.1.4.1 Interfaces

- Major functions must be accessible from both Web GUI and programmatic Web Service clients.
- Interface will accept a *projection* data object and return a *pathway* data object.
- Interface will accept a *pathway* data object and return a *pathway validator* data object.

5.1.4.2 Efficiency

- Data handling needs to be compatible with HPC data management standards to facilitate efficient use of storage and I/O resources. Data objects will be internally stored using Matlab data structures and written out using HDF5 format.
- The computations for the pipeline that translates a projection data object into a pathway data object should be efficient enough to allow rapid prototyping and validation.

5.1.4.3 Reliability

- The NIP is expected to have regular (2 / month) scheduled maintenance windows. Each maintenance window will be no more than 30 minutes long.
- Individual non-Platform services on which the Platform depends may be more unreliable, but the Platform will provide a service status page that will show the status of the services on which it depends.

5.1.4.4 Monitoring

The NIP should be monitored by an operations monitoring suite. This suite will report on usage, performance and health metrics.

5.1.5 Brain Addressing System: Software

- *Projection* data object interpreter
- Cell Index generator
- Voxel distribution generator
- Element distributor
- Pre and postsynaptic element pairing tool
- Morphology constructor
- *Pathway* data object builder
- *Pathway* data object inspector
- Synaptic statistics module
- *Pathway* data object validator
- *Pathway validation* data object builder



5.1.6 Brain Addressing System: Physical Architecture

This section describes the components that make up the Brain Addressing System, and how they interact with the other pieces of the NIP.

All of the components are accessed through a network-accessible service API. However, not all components will be exposed to end-users of the Neuroinformatics Portal. Some will be reserved for internal use.

5.1.6.1 Components

- *Projection* data object interpreter
- Cell Index generator (sending and receiving)
- Presynaptic/postsynaptic element voxel distribution generator
- Presynaptic/postsynaptic element distributor
- *Pathway* data object builder
- *Pathway* data object inspector
- *Pathway* data object validator
- Pre and postsynaptic element pairing tool
- Morphology constructor
- Synaptic statistics module
- *Pathway validation* data object builder

5.1.7 Brain Addressing System: Services required from other Platforms

- HPC - Job Management
- HPC - Data transfer
- HPC - Data storage
- HPC - Fast optimisation routines
- Neuroinformatics - search services
- Neuroinformatics - curated brain atlases
- Neuroinformatics - atlas spatial conversion tools

5.1.8 Brain Addressing System: Services provided to other Platforms

- Brain connectivity configuration and building
- Brain connectivity validation and analysis

5.1.9 Brain Addressing System: Prerequisites

5.1.9.1 Required

- Matlab for design and development of pilot versions of algorithm.
- NIP interface to (compiled) Matlab functions.



- Function that “voxelises” target brain areas.
- Robust and efficient implementation of machine learning algorithms in Matlab and C.
- Fast implementation of weighted minimum spanning tree algorithm.
- Pilot anatomical data to implement use case for validation of Brain Addressing System.
- HPC Platform and service development.
- NIP and service development.

5.1.9.2 Preferred

- Brain Simulation Platform

5.1.10 Brain Addressing System: Necessary Parallel Activities

The Brain Addressing System needs Projection data objects that are constrained by experimental data. Therefore we need data of SP1 for cell and synapse densities as well as the morphology of long-range axonal projections. In parallel, data from external sources is necessary, such as the Allen Institute for Brain Science, mice connectome and DTI measurements from rodents (Allan Johnson at Duke University), as well as data from emerging techniques such as GRASP, which stands for GFP reconstruction across synaptic partners.

5.1.11 Brain Addressing System: Hardware/Software Functions

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.1	Leader:	Paul Tiesinga
Function Name:	<i>Projection</i> data object interpreter		
Test Case:	Successful translation of <i>Projection</i> data object interpreter into internal Matlab data structure used by subsequent functions in the pipeline		
Planned Start Date:	July 2014	Planned Completion Date:	Aug 2014
Requires Functions:			

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.2	Leader:	Paul Tiesinga
Function Name:	Cell Index generator		
Test Case:	Successful generation of number of cells specified by the <i>Projection</i> data object into internal Matlab data structure used by subsequent functions in the pipeline		
Planned Start Date:	August 2014	Planned Completion Date:	September 2014
Requires Functions:	5.4.1.1		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.3	Leader:	Paul Tiesinga
Function Name:	Voxel distribution generator		
Test Case:	Successful ingestion of voxelisation of the postsynaptic brain area into internal Matlab data structure and indexed pre- or postsynaptic cells specified by the Cell Index generator and subsequent generation of the probability of an pre or -post synaptic element for each voxel.		
Planned Start Date:	September 2014	Planned Completion Date:	October 2014
Requires Functions:	5.4.1.1 and 5.4.1.2		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.4	Leader:	Paul Tiesinga
Function Name:	Element distributor		
Test Case:	Successful ingestion of output of voxel distribution generator and the generation for each voxel of the realised post or presynaptic element		
Planned Start Date:	October 2014	Planned Completion Date:	December 2014
Requires Functions:	5.4.1.1, 5.4.1.2 and 5.4.1.3		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.5	Leader:	Paul Tiesinga
Function Name:	Pre and postsynaptic element pairing tool		
Test Case:	Successful ingestion of pre and postsynaptic elements generated by the element distributor and the generation of pairs of pre- and postsynaptic elements for all elements.		
Planned Start Date:	December 2014	Planned Completion Date:	February 2015
Requires Functions:	5.4.1.1, 5.4.1.2 and 5.4.1.4		



Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.6	Leader:	Paul Tiesinga
Function Name:	Morphology constructor		
Test Case:	Successful ingestion of pre or postsynaptic elements generated by the element distributor (and prior generated elements); the generation of a morphology using these elements as seed points optimising a weighted sum of wire length and path length.		
Planned Start Date:	February 2015	Planned Completion Date:	March 2015
Requires Functions:	5.4.1.1, 5.4.1.2 and 5.4.1.4		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.7	Leader:	Paul Tiesinga
Function Name:	<i>Pathway</i> data object builder		
Test Case A:	Successful ingestion of output of the Pre and postsynaptic element pairing tool and the Morphology constructor. Generation of the pathway, specified by for each synapse, the identity of presynaptic cell and axon branch by which it is made and the identity of the postsynaptic cell and the dendritic branch on which it is made.		
Test Case B	Use case: select a <i>projection</i> data object and turn it into a pathway data object		
Planned Start Date:	March 2015	Planned Completion Date:	July 2015
Requires Functions:	5.4.1.1, 5.4.1.2, 5.4.1.5 and 5.4.1.6		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.8	Leader:	Paul Tiesinga
Function Name:	<i>Pathway</i> data object inspector		
Test Case:	Successful ingestion of a <i>Pathway</i> data object and extraction of components for other functions that validate or statistically characterise the pathway		
Planned Start Date:	July 2015	Planned Completion Date:	October 2015
Requires Functions:	5.4.1.7		



Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.9	Leader:	Paul Tiesinga
Function Name:	Synaptic statistics module		
Test Case:	Successful ingestion of a <i>Pathway</i> data object and extraction of relevant components via <i>Pathway</i> data object inspector. Determination of the distribution of branch order for the pre and post synaptic elements. [Additional functionality will be added]		
Planned Start Date:	October 2015	Planned Completion Date:	December 2015
Requires Functions:	5.4.1.1, 5.4.1.7, 5.4.1.8		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.10	Leader:	Paul Tiesinga
Function Name:	<i>Pathway</i> object validator		
Test Case:	Successful ingestion of a <i>Projection</i> data object and <i>Pathway</i> data object and extraction of relevant components via <i>Pathway</i> data object inspector and determination of their pertinent statistics via Synaptic statistics module.		
Planned Start Date:	December 2015	Planned Completion Date:	Feb 2015
Requires Functions:	5.4.1.1, 5.4.1.7, 5.4.1.9		

Task No:	5.4.1	Partner:	SKU
Function No:	5.4.1.11	Leader:	Paul Tiesinga
Function Name:	<i>Pathway validation</i> data object builder		
Test Case:	Successful ingestion of the output of <i>Pathway</i> object validator and packaging of the outcome of statistical comparisons between desired and generated statistics for use in summary functions		
Planned Start Date:	Feb 2015	Planned Completion Date:	Mar 2015
Requires Functions:	5.4.1.1, 5.4.1.7, 5.4.1.10		



6. Brain Atlases (WP5.5)

The HBP will use the tools just described to build multi-level atlases of the mouse and human brains (T5.5.1 and T5.5.2, respectively). The design will encourage research groups outside the Project to deposit data in the atlases, enabling global collaboration to integrate data across scales in a single atlas for each species.

6.1 The Mouse Brain Atlas (T5.5.1)

T5.5.1 will use the Atlas Builder from T5.1.6 (Section 2.2) to build multi-level atlases of the mouse brain. It will federate high-quality data for gene expression, protein maps, immunohistochemical staining, electrophysiology, DTI, tract tracing and more. T5.5.1 will annotate the data, and assign it to the common coordinate space, using standard data models to make the data accessible through public portals and software APIs. It will also provide means for community review, annotation and curation.

The Atlas Builder requires access to 3D template packages as frameworks for data analysis and integration.

6.1.1 The Mouse Brain 3D Template Package

The Mouse Brain 3D template package will consist of a reference coordinate space and parcellations for the adult Black6 mouse strain and a standard ontology of brain areas and structures.

The Mouse Brain 3D template package contains:

- Co-registered volumetric representations of Nissl stack (NIfTI file, 528x320x456 voxels, resolution is 25 µm isotropic).
- Atlas delineations of about 600 structures organised into 82 higher-level structures (NIfTI file, 528x320x456 voxels, resolution is 25 µm isotropic).
- Metadata descriptor file.
- Smoothed non-intersecting meshes generated from the atlas delineations (STL files).

The data to construct the 3D template package were acquired from <http://help.brain-map.org/display/mousebrain/API>. ABA reference space is provided. Conversion to standardised Waxholm space is available.

6.1.2 The Rat Brain 3D Template Package

A Rat Brain 3D template package, similar to the one provided for mouse, will be provided.

The Rat Brain 3D template package contains:

- MRI: T2*-weighted, 512 x 512 x 1024 voxels, 39 microns isotropic, 32-bit floating point format NIfTI.
- DTI: originals acquired at 256 x 256 x 512 voxels (78 microns isotropic), and resampled to 512 x 512 x 1024 voxels (resampled without interpolation to 39 microns isotropic) NIfTIs.
- b0 image (T2- weighted MRI) - 16-bit signed integer NIfTI.
- FA, colour FA, DWI, ADC - 16-bit signed integer, 24-bit RGB, 16-bit signed integer, 16-bit signed integer NIfTIs respectively.



- Atlas delineations of primary brain regions, about 76 regions in the first release, 512 x 512 x 1024 voxels, 39 microns isotropic, in 16-bit indexed unsigned integer NIfTI and the brain mask to hide non-brain structures.
- Smoothed non-intersecting STL meshes generated from the atlas delineations.
- Metadata descriptor file.

All volumes (of the same sizes) are co-registered. Conversion from Waxholm space to stereotaxic (skull based) coordinates is provided. Initial data are shared and published under a Creative Commons License at the INCF Software Center (<http://software.incf.org>) with reference to the original article.

6.1.3 Rodent Brain Image Data Collections

Collections of experimental image data and normal material from mouse and rat brain are available and will be anchored to the 3D templates described above, using the infrastructure, tools, and procedures outlined under Brain Atlas Builder (T5.1.6), Section 2.2. The data collections consist of series of high-resolution section images of histological sections, typically covering the whole brain with high sampling intensity. The experimental data demonstrate brain-wide distribution of labels/markers at subcellular, cellular or regional levels, representing genes, molecules, circuits or other features in normal material or in genetic disease models. The normal material provides information on cyto- and myeloarchitecture.

The collections are:

- The spatial distribution of **glutamate transporter protein in the rat brain** immunohistochemically visualised using antibodies anti-GLT1a and anti-GLT1b in adult Wistar rat, with neighbouring sections stained for myeloarchitecture and cytoarchitecture.
 - Data from: Holmseth S, Scott HA, Real K, Lehre KP, Leergaard TB, Bjaalie JG, Danbolt NC. [The concentrations and distributions of three C-terminal variants of the GLT1 \(EAAT2; slc1a2\) glutamate transporter protein in rat brain tissue suggest differential regulation.](#) *Neuroscience* 2009; 162:1055-71
- The spatial distribution of **glutamate transporter protein in the mouse brain** immunohistochemically visualised using antibodies anti-GLT1a or anti-GLT1b in an adult C57Bl9 mouse, with neighbouring sections stained for myeloarchitecture and cytoarchitecture.
 - Data from: Holmseth S, Scott HA, Real K, Lehre KP, Leergaard TB, Bjaalie JG, Danbolt NC. [The concentrations and distributions of three C-terminal variants of the GLT1 \(EAAT2; slc1a2\) glutamate transporter protein in rat brain tissue suggest differential regulation.](#) *Neuroscience* 2009; 162:1055-71
- The spatial distribution of the **neurotransporter VGLUT1** immunohistochemically visualised in an 8-week old adult male Wistar rat using a polyclonal rabbit antibody against VGLUT1.
 - Stensrud MJ, Chaudry FA, Leergaard TB, Bjaalie JG, Gundersen V. VGLUT3 in the rodent brain: vesicular co-localisation with VGAT *J Comp Neurol*, 2013, doi: 10.1002/cne.23331
- The brain-wide distribution of **axonal connections from the primary somatosensory cortex** in rats mapped using axonal tract tracing techniques.



- Data from: Zakiewicz IM, Bjaalie JG, Leergaard TB. Brain-wide map of efferent projections from rat barrel cortex. *Frontiers in Neuroinformatics* 2014, 8: Article 5:1 & Zakiewicz IM, van Dongen YC, Leergaard TB, Bjaalie JG. Workflow and atlas system for brain-wide mapping of axonal connectivity in rat. *PLoS ONE*, 2011;6(8):e22669. doi: 0.1371/journal.pone.0022669
- The **cyto- and myeloarchitecture of the normal rat brain** mapped using high-angular resolution diffusion MRI, and histologically visualised in coronal and sagittal histological sections.
 - Data from: Leergaard TB, White NS, De Crespigny A, Bolstad I, D'Arceuil H, Bjaalie JG, Dale AM: On resolving complex myeloarchitectures with diffusion magnetic resonance imaging. *PLoS ONE* 2010; 5: e8595. doi:10.1371/journal.pone.0008595 & White NS, Leergaard TB, D'Arceuil H, Bjaalie JG, Dale AM. Probing tissue microstructure with Restriction Spectrum Imaging: Histological and theoretical validation. *Human Brain Mapping*, 2013, 34:327-46. doi: 10.1002/hbm.21454
- **Neurodegenerative changes in basal ganglia neural networks** occurring in aged transgenic rat model of **Huntington disease**, measured using diffusion kurtosis MRI and immunohistochemical visualisation of glial cells.
 - Data from: Blockx I, Verhoye M, Van Audekerke J, Bergwerf I, Kane JX, Delgado Y, Palacios R, Veraart J, Jeurissen B, Raber K, von Hörsten S, Ponsaerts P, Sijbers J, Leergaard TB, Van der Linden A. *Neuroimage* 2012, 1;63(2):653-62. doi: 10.1016/j.neuroimage.2012.06.032 & Antonsen BT, Jiang Y, Veraart J, Qu H, Nguyen HP, Sijbers J, von Hörsten S, Johnson GA, Leergaard TB. Altered diffusion tensor imaging measurements in aged transgenic Huntington disease rats. *Brain Struct Funct*. 2013, 218(3):767-78. doi: 10.1007/s00429-012-0427-0.
- The spatial distribution of the **tetracycline-responsive prion protein (PrP) promoter** histologically visualised in PrP/LacZ Tet-Off promoter mice.
 - Data from: Boy J, Leergaard TB, Schmidt T, Odeh F, Bichelmeier U, Nuber S, Holzmann C, Wree A, Prusiner SB, Bujard HB, Riess O, Bjaalie JG. Expression mapping of tetracycline-responsive protein promoter: digital atlasing as a basis for generating cell-specific disease-models. *NeuroImage* 2006; 33: 449-462
- The spatial distribution of the **tetracycline-responsive calmodulin-dependent protein kinase II (CamKII) promoter** histologically visualised in CamKII/LacZ Tet-Off promoter mice.
 - Data from: Odeh F, Leergaard TB, Boy J, Schmidt T, Riess O, Bjaalie JG. Atlas of transgenic Tet-Off Ca²⁺/calmodulin-dependent protein kinase II and prion protein promoter activity in the mouse brain. *NeuroImage*, 2011, 54:2603-11
- The spatial distribution of the **tetracycline-responsive neuropsin (Nop) promoter** histologically visualised in Nop-tTA mice.
 - Yetman et al., material from Joanna Jankowsky, Baylor College of Medicine, Houston, USA)
- The spatial distribution and variance of amyloid-beta deposits in aged double-transgenic mice carrying the Arctic and Swedish amyloid- β precursor protein.
 - Data from: Lillehaug S, Syverstad GH, Nilsson LNG, Bjaalie JG, Leergaard TB, Torp R. Brainwide distribution and variance of amyloid-beta deposits in tg-ArcSwe mice. *Neurobiology of Aging* 2014, 35:556-564.

6.1.4 Mouse Brain Atlas: Hardware/Software Functions

Task No:	5.5.1	Partner:	UIO
Function No:	5.5.1.1	Leader:	Jan Bjaalie
Function Name:	Validate atlas parcellations		
Test Case:	Validation of the shared atlas delineations / parcellations. Results of testing and community feedback will be used constantly over the task period to modify and expand the atlas delineations / parcellations.		
Planned Start Date:	June 2014	Planned Completion Date:	September 2015
Requires Functions:	Existing 3D atlas templates. Existing tools for delineations of structures.		

Task No:	5.5.1	Partner:	UIO
Function No:	5.5.1.2	Leader:	Jan Bjaalie
Function Name:	Create atlases based on normal histological material		
Test Case:	Anchor / register series of 2D images of histological sections from normal material to 3D atlas template, using approach outlined in SP5-UC-011.		
Planned Start Date:	September 2014	Planned Completion Date:	March 2015
Requires Functions:	5.1.6.1		

Task No:	5.5.1	Partner:	UIO
Function No:	5.5.1.3	Leader:	Jan Bjaalie
Function Name:	Create atlases based on disease model histological material		
Test Case:	Anchor / register series of 2D images of histological sections from disease model material to 3D atlas template, using approach outlined in SP5-UC-011.		
Planned Start Date:	January 2015	Planned Completion Date:	Sept 2015
Requires Functions:	5.1.6.1		

Task No:	5.5.1	Partner:	UIO
Function No:	5.5.1.4	Leader:	Jan Bjaalie
Function Name:	Query of atlases		
Test Case:	Semantic and spatial query of primary data and metadata in the atlases		
Planned Start Date:	April 2015	Planned Completion Date:	March 2016
Requires Functions:	5.5.1.2, 5.5.1.3		

The functions under 5.5.1, Mouse Brain Atlas, build on the:

- 3D atlas templates already produced in the early phase of the Project (see report for Mo 7 - 9).



- The use of standard tools, such as ITK Snap, for introducing atlas delineations / parcellations on the 3D atlas templates.
- The AligNII tool as described in SP5-UC-011, allowing anchoring / registration of 2D images to the 3D atlas templates, thereby assigning spatial coordinates to each image.
- The infrastructure we have available at UIO for data management, with support for data uploading and downloading, storage and use of 3D atlas templates and parcellations of choice in combination with experimental data uploaded in the system.

6.2 The Human Brain Atlas (T5.5.2)

T5.5.2 will use the Atlas Builder from T5.1.6 to build a multi-level atlas of the human brain. The atlas will contain different reference brains and parcellations for the human brain and a standard ontology of brain areas and structures (Amunts K, Hawrylycz, M.J.; Van Essen, D.C., Van Horn, J.D., Harel, N., Poline, J-B., De Martino, F., Bjaalie, J.G., Dehaene-Lambertz, G., Dehaene, S., Valdes-Sosa, P., Thirion, B., Zilles, K., Hill, S., Abrams, M.B., Tass, P.A., Vanduffel, W., Evans, A.C., Eickhoff, S.B; Interoperable Atlases of the Human Brain, Frontiers, submitted). T5.5.2 will federate high-quality data for cellular architecture, gene expression, fMRI, DTI, tract tracing, high-resolution fibre architecture based on polarised light imaging (PLI) with subsequent tractography, receptor architecture of most relevant classical neurotransmitter systems, and more. Data will come from SP2 (Strategic Human Brain Data) as well as from external partners (Allen Brain Atlas). T5.5.2 will annotate the data, and define transformation rules between the different available reference brains. T5.5.2 will assign the data to the common coordinate space using standard data models to make the data accessible through public portals and software APIs. It will also provide means for community review, annotation and curation.

Human Brain Atlas requirements are as follows:

6.2.1 *BigBrain Atlas for High-Resolution Microscopical Data*

TIFF files published under (<https://bigbrain.loris.ca/main.php>) and with reference to original article) Amunts K, Lepage C, Borgeat L, Mohlberg H, Dickscheid T, Rousseau M, Bludau S, Bazin PL, Lewis LB, Oros-Peusquens AM, Shah NJ, Lippert T, Zilles K, Evans AC (21-6-2013) BigBrain: An ultrahigh-resolution 3D human brain model. Science, 340(6139): 1472-1475).

3D-reconstructed histological data set:

7,404 original images acquired at 10 x 10 pixel in-plane physical resolution, 20 micron section thickness (12,000 pixel x 10,000 pixel in dependence on size), down-sampled to 20 microns isotropic, in Tiff-format, then converted Minc-format.

3D reconstructed histological sections: 7,000 x 6,000 pixels in-plane in PNG-format; 12 MByte due to compression per image.

MRI: T1-weighted, 0.4 x 0.4 x 0.8 mm, 6 averages, MPRAGE of the formalin-fixed brain

Atlas delineations of cytoarchitectonically defined brain regions:

Cortical areas and nuclei, currently about 100 regions in the first release, represented in the single subject T1 weighted MNI reference space ("Colin space") at 1 x 1 x 1 mm, in 32-bit floating points (for delineations) and 8 bit (for the Colin space).



Every structure of each hemisphere represents a single NIfTI -1 format in this reference space. The structures are based on observer-independent delineations of areas in 10 human *post mortem* brains according to the original publications (e.g., Caspers J, Zilles K, Eickhoff SB, Schleicher A, Mohlberg H, Amunts K (2013) Cytoarchitectonical analysis and probabilistic mapping of two extrastriate areas of the human posterior fusiform gyrus. *Brain Struct.Funct.*, 218: 511-526).

In addition, all structures are projected to the surface of the single subject Colin space as object files and labels. Data are publicly available (<https://www.jubrain.fz-juelich.de/apps/cytoviewer/cytoviewer-main.php> and http://www.fz-juelich.de/inm/inm-1/DE/Forschung/_docs/SPMANatomyToolbox/SPMANatomyToolbox_node.html).

A transformation between the BigBrain data set and the Colin space at a resolution of maximally down to 0.1 mm has been provided.

Future versions will include, in addition to Cytoarchitectonic data, data from other aspects of brain organisation, labels (e.g., gyri, sulci), as well as metadata including cell densities, receptor concentrations and more.

6.2.2 Montreal Neurological Institute Reference Space (MNI space)

The MNI was developed as a series of different stereotactic reference systems, which allow neuroscientists to combine data from many subjects such that group-averaged signals, be they structural or functional, can be detected. The neuroimaging community frequently uses these reference spaces. Spaces are publicly available (<http://www.bic.mni.mcgill.ca/ServicesAtlases/HomePage>).

The concept is according to Evans AC, Janke AL, Collins DL, Baillet S: Brain templates and atlases *Neuroimage*. 2012, 62(2):911-22.

The templates are based on MR-images of human subjects, either as group averages (MNI-ICBM 152, MNI 305) or as the individual T1 weighted template of the MNI ("Colin space" or MNI Colin27). The first two templates are the result of averaging individual MR brain data sets, and they exist in several version.

6.2.3 Allen Human Brain Atlas (AHBA)

Core data:

A multi-modal, multi-resolution atlas detailing gene expression across the adult human brain is provided. Complete microarray data sets for the full complement of six brains are available for download from <http://human.brain-map.org>.

The data sets contain gene expression values normalised across all brains, as well as probe and sample metadata necessary for analysis. Normalised data values are accessible in multiple ways in the AHBA: (1) through displayed data values and 'download this data' links in the interactive web-based application; (2) the application programming interface (API); and (3) downloadable .csv files, including archived files for historical array data processed through the original normalisation strategy.

Approximately 500 anatomically discrete samples per hemisphere were collected from cortex, subcortex, cerebellum and brainstem of each brain and profiled for genome-wide gene expression using a custom Agilent 8x60K cDNA array chip.

Human brain atlas content:

Collections of human brain image data demonstrating spatial distributions of cellular and molecular markers in adult human brains, as well as *in vivo* diffusion connectivity data, fMRI data from functional localisers, and ultra-high resolution *post mortem* data on the fibre architecture (3D-PLI data).

- The spatial segregation of the cerebral cortex and subcortical nuclei based on **cytoarchitectonic analyses** in ten human *post mortem* brains.

6.2.4 Data from the JuBrain Atlas

(<https://www.jubrain.fz-juelich.de/apps/cytoviewer/cytoviewer-main.php> and http://www.fz-juelich.de/inm/inm-1/DE/Forschung/_docs/SPMANatomyToolbox/SPMANatomyToolbox_node.html)

Concept: Zilles K, Amunts K (2010) Centenary of Brodmann's map - conception and fate. *Nature Reviews Neuroscience* 11(2): 139-145.

- The spatial distribution of **receptor distributions of neurotransmitter systems** (e.g., glutamatergic AMPA, kainite and NMDA receptors, acetylcholinergic muscarinic M1 and M2, cholinergic nicotinic receptors, noradrenergic α_1 and α_2 , serotonergic 5-HT_{1A} and more and 5-HT_{2A}) processed and quantified using receptor autoradiography, including their receptor fingerprints expressing the balance of the different receptor types. Data from: Zilles K, Amunts K (2009) Receptor mapping: Architecture of the human cerebral cortex. *Current Opinion in Neurology* 22(4): 331-339.
- The **fibre architecture** as defined using 3D-PLI. Data from: Axer M, Graessel D, Kleiner M, Dammers J, Dickscheid T, Reckfort J, Hütz T, Eiben B, Pietrzyk U, Zilles K, Amunts K (2011) High-resolution fibre tract reconstruction in the human brain by means of three-dimensional polarised light imaging (3D-PLI). *Frontiers in Neuroinformatics*, 5: 34.
- The **gene expression data** mapped in six human brains from the Allen Institute <http://human.brain-map.org/>. Data from: Hawrylycz MJ, Lein ES, Guillozet-Bongaarts AL, Shen EH, Ng L, Miller JA, van de Lagemaat LN, Smith KA, Ebbert A, Riley ZL, Abajian C, Beckmann CF, Bernard A, Bertagnolli D, Boe AF, Cartagena PM, Chakravarty MM, Chapin M, Chong J, Dalley RA, Daly BD, Dang C, Datta S, Dee N, Dolbeare T, Faber V, Feng D, Fowler DR, Goldy J, Gregor BW, Haradon Z, Haynor DR, Hohmann JG, Horvath S, Howard RE, Jeromin A, Jochim JM, Kinnunen M, Lau C, Lazarz ET, Lee C, Lemon TA, Li L, Li Y, Morris JA, Overly CC, Parker PD, Parry SE, Reding M, Royall JJ, Schulkin J, Sequeria PA, Slaughterbeck CR, Smith SC, Sodt AJ, Sun. [An anatomically comprehensive atlas of the adult human brain transcriptome](#) *Nature*, December 2012.

7. Roadmap for Brain Data Integration and Navigation

Month	Data sets	Responsible
18	Load mouse template atlas and navigate brain regions- Allen Mouse Atlas	Sean Hill
18	Integrate predicted cell densities (neuronal, glial, cell types)	Sean Hill
18	Integrate predicted synapse projections/densities	Sean Hill Paul Tiesinga
18	Text-mined literature - Find literature related to each brain area from Allen Brain ontology	Sean Hill
18	76 brain regions in rat Waxholm Space defined based on high resolution MRI and DTI	Jan Bjaalie
24	Combined cyto- and myeloarchitecture, rat, Waxholm Space	Jan Bjaalie
24	16 subdivisions of the hippocampus in rat Waxholm Space defined based on MRI, DTI, cyto- and chemoarchitecture	Jan Bjaalie
24	Integrate rat neocortical microcircuitry data (neuron morphologies, electrophysiology, gene expression data)	Sean Hill
24	Muscarinic M2 receptor distributions, rat, 3D reconstruction from ~500 sections, Waxholm space	Karl Zilles
24	~100 cytoarchitectonically defined brain regions in MNI Colin27 space	Katrin Amunts
24	Integrate text-mined literature - mouse region-region connectivity	Sean Hill
24	BigBrain template, human, including transformation parameters from and to MNI Colin27 space (at ~1mm resolution)	Katrin Amunts
24	Anatomical probability maps from SPM toolbox in MNI Colin27 space	Simon Eickoff
24	Connections of SI barrel cortex, rat, 3D reconstruction from ~150 sections, Waxholm space	Jan Bjaalie
30	Tetracycline-responsive neuropsin (Nop) promoter distribution, mouse, ABI space, ~100 sections	Jan Bjaalie
30	Amyloid-beta deposit distribution in Swe/Arc mouse, ABI space, ~100 sections	Jan Bjaalie
30	~100 cytoarchitectonically defined brain regions in MNI Colin27 space	Katrin Amunts
30	3D Polarised Light Imaging fibre orientation maps, rat, Waxholm space, 60x64x64 μ m	Markus Axer
30	Load infant human brain template	Ghislaine Dehaene

Figure 10: Roadmap for brain data integration and navigation

8. Key Performance Indicators (KPIs)

Task	KPI	ID	Target values			
			M12	M18	M24	M30
5.1.2	Number of completed functions	SP5-SKPI-001			1	2
5.1.4	Percentage of data covered for current data provider	SP5-SKPI-002	50%	100%		
5.1.6	Number of completed functions	SP5-SKPI-003		1	2	
5.3.1	Number of completed functions	SP5-SKPI-004	1		2	3
5.3.2	Number of completed functions	SP5-SKPI-005	1	2		3
5.3.3	Number of completed functions	SP5-SKPI-006	1	2		3
5.3.4	Number of completed functions	SP5-SKPI-007	1	2		3
5.3.5	Number of completed functions	SP5-SKPI-008	1	2		3
5.3.6	Number of completed functions	SP5-SKPI-009	1	2		3
5.3.7	Number of completed functions	SP5-SKPI-010		1	2	3
5.3.8	Number of completed functions	SP5-SKPI-011		1	2	3
5.3.9	Number of completed functions	SP5-SKPI-012		1		3
5.3.10	Number of completed functions	SP5-SKPI-013	1			2
5.3.11	Number of completed functions	SP5-SKPI-014			1	2
5.4.1	Number of completed functions	SP5-SKPI-015	3	8	10	11
5.4.2	Number of completed functions	SP5-SKPI-016	4	6	9	11
5.5.1	Number of completed functions	SP5-SKPI-017		1	3	4
5.6.1	Number of completed functions	SP5-SKPI-018		3	4	

Figure 11: Key Performance Indicators (KPIs)

9. Glossary

2D image	High-resolution bitmap image, currently supported formats: TIFF, JPEG, PNG; work in progress: Yokogawa CV1000, CZI. The list of the supported formats relies on OME Bio-Formats library .
3D data	Coming from a soma confocal microscopy image contains 3D points and the edges that join them (the faces). This data contains the 3D representation of the soma that is computed and statistically analysed to obtain the quantitative characterisation.
3D stack	A series of 2D consecutive images from a tissue sample or preparation obtained via electron or light microscopy.
3D template package	A general 3D template package consists of an MRI template (NIfTI file), a delineations template (NIfTI file, produced from MRI template) and meshes (STL files, produced from delineations template).
AlignII	Browser-based anchoring tool.
Anchoring	The process of positioning (translate, rotate, scale) 2D images in 3D space defined by a 3D template package. Through region-by-region anchoring, several positions could be stored for each 2D image.
Annotations	Vector drawings (SVG) superimposed on 2D images.
Brain addressing system	Builds connectivity matrices. Neurons in the source area send out axons that terminate on the dendrites and soma of neurons in the target area. Each neuron has dendrites and an axon, both of which can be described in terms of a branching structure. An instantiated synaptic connection is characterised by the index of the sending neuron, the index of receiving neuron, the index of axonal branch of the sending neuron and index of the dendritic branch of the receiving neuron between which the synapse is made.
Brain atlas	A collection (usually a stack) of 2D images anchored to a 3D template package.
Complete spatial randomness (CSR)	AKA homogeneous Poisson process. 'Reference' or 'benchmark' model of a random point pattern.
Converter	A server-side process of creating pyramids, previews and thumbnails from 2D images.
Data file	A file that contains data common to electrophysiological recordings. This includes in particular, but is not limited to, spike time data, discretely sampled continuous signal data (such as LFP or measures of the stimulus), and trigger events. In the context of the Collaboratory, data files represent an artefact.
Delineations	Boundaries of structures in the brain shown in the context of brain atlas space.
Dendritic basal arbour	The part of the neuron that includes the general features of the neuron observed from the apical dendrite, such as the layout of the basal dendritic trees and their features.
Dendritic & soma data	Contains properties of dendritic specification and quantitative characterisation of the soma.
F function	Cumulative distribution function of the empty space distance.
G function	Cumulative distribution function of the nearest-neighbour distance for a typical point in the pattern.

HDF5	A hierarchical file format designed to store numerical data.
Hybrid Bayesian network	A Bayesian network that combines continuous, discrete and in this case, also angular variables.
K function	Ripley's function, expected number of other points of the process within a distance r of a typical point of the process divided by the intensity.
L function	Commonly used transformation of K function making visual assessment of the graph much easier.
Local field potential (LFP)	The continuous low-pass filtered signal from an extracellular recording electrode (high cut of the filter typically in the range of 100-500 Hz).
Metadata	The data available and recorded, either in an automated fashion or by the user, before, during and after an experiment providing additional information relevant in interpreting and relating the electrophysiological and behavioural recordings.
Optimisation data	Contains the data corresponding to a 3D soma shape generated by simulation.
PAB	Metadata in the hierarchical form: Project -> Animal(s) -> Block(s).
Pathway data object	Contains an instantiation of a connection matrix.
Pathway validation data object	Contains the validation data of a pathway data object
Preview	A downscaled copy of 2D image.
Projection data object	Contains the statistics of the projection between two areas and can be accessed in the atlas by referring to the sending as well as receiving area.
Random sequential adsorption (RSA)	Spatial process where the pattern is constructed by placing spheres in three-dimensional space iteratively and randomly, with their radii following a probability density function. If any newly generated sphere intersects with an existing sphere, the new sphere is rejected and another sphere with a different centre and radius is generated. This process is stopped when the required number of spheres is reached.
Sample	The original tissue preparation from which a 3D stack is obtained.
Segmentation	A set of different segmented structures.
Segmented structure	A set of 3D voxels corresponding to a biological structure (either cell or sub-cell entities) annotated with its corresponding semantic information.
Soma	Represented by several morphological characteristics. Somas are described by a set of statistical distributions which represent their main properties as height, width, volume, etc.



Spatial point pattern	Realisation of a spatial point process.
Spatial point process	Mathematical model that describes the arrangement of objects irregularly or randomly distributed in space forming patterns.
Spike	The action potential of a neuron and its time point of the firing. Spikes are measured by an extracellular recording electrode in the vicinity of a neuron. Spikes are detected as a threshold crossing-event on the electrode signal that displays a typical spike waveform (to distinguish it from recording artefacts).
Spike sorting	The process of assigning labels to the individual spikes in an electrophysiological recording in order to group those spikes, which are likely to have originated from a single neuron. Labels are assigned on the basis of the similarity of observed spike waveforms in the analogue signal.
Stereological counting frame/counting frame	Inclusive/exclusive boundaries of the 3D stack to be considered in the counting of segmented structures.
Synaptic apposition surface (SAS)	The surface between the active zone and the post-synaptic density representing the area of the synaptic junction.
Template	Data set specifying a brain atlas space, i.e., a 3-D coordinate system of the brain.