Open Mind	OMNI Summit Microservice V1.X Project Plan	Doc. Number	Rev.
		OMNI-SM-001	**1**

Contents

1 Document Purpose

This document provides a comprehensive development plan for OMNI Summit Microservice V1. This plan defines project objectives, project organization, resources, activities, sequence of activities, documentation deliverables, and a list of finished device deliverables in order to provide visibility and control of the execution of the project.

2 Document Scope

This document serves as the primary planning document for OMNI Summit Microservice V1, integrating the planning required for project management, software development, and documentation creation. This plan controls additional activities through the use of subordinate planning documents, including a design (software) verification plan.

3 Project Background

Over the last decade, the field of neuromodulation research has been dramatically accelerated by the rising availability of human-use implantable research device platforms. Deep Brain Stimulation (DBS) research, in particular, is seeing a proliferation of new investigational devices for use in the field, with more on the way, including the Activa PC+S [1] and Summit RC+S [2], the DyNeuMo Mk-1 [3], and the Brain InterChange System [4]. These research-focused Implantable Neuro-Stimulators (INSs) allow researchers to develop custom computer software for device interaction using manufacturer-provided software development kits (SDKs). Published research enabled through this “SDK-driven” strategy includes studies exploring adaptive deep-brain stimulation (aDBS) for PD [5], [6], essential tremor [7], epilepsy [8], and neuropsychiatric illnesses [9]. However, these first-generation research SDKs are tightly integrated with the INS hardware, inhibiting code reuse across studies. The result of this tightly coupled software/hardware approach is that research software applications are written as monolithic applications, with the device-specific research SDKs called directly from the software, making it difficult to separate reusable code from protocol-specific implementations.

The proliferation of investigational devices creates a fragmented landscape of programming languages and operating-system support, posing significant barriers to future research support. The Medtronic family of INS hardware is illustrative of this problem: whereas the Activa PC+S platform provided a Java SDK for Windows, the next generation hardware, Summit RC+S, moved from Java to C# for the SDK. This demonstrates that even within the same family of INS devices from the same manufacturer, there is limited opportunity for interoperability - each new device requires rewriting most software from scratch.

The landscape of INS hardware devices is further complicated by their unique capabilities. To perform research with new devices, researchers must often translate configuration parameters from a device used to pilot prior work, as INS platforms frequently lack one-to-one mappings for configuration options. There are long-term impacts of these limitations as well. As the field of adaptive neuromodulation matures, replication studies will become increasingly important -- but owing to INS hardware evolution these studies may require rewriting research software to target new device hardware. It is anticipated such rewrites will often necessitate a translation between programming languages, as well as a translation of INS configuration options. Previous discussion of the scientific and technical concerns presented by a fragmented research tools space have highlighted the need for a platform-based approach to solving what will become a key barrier for future work [10].

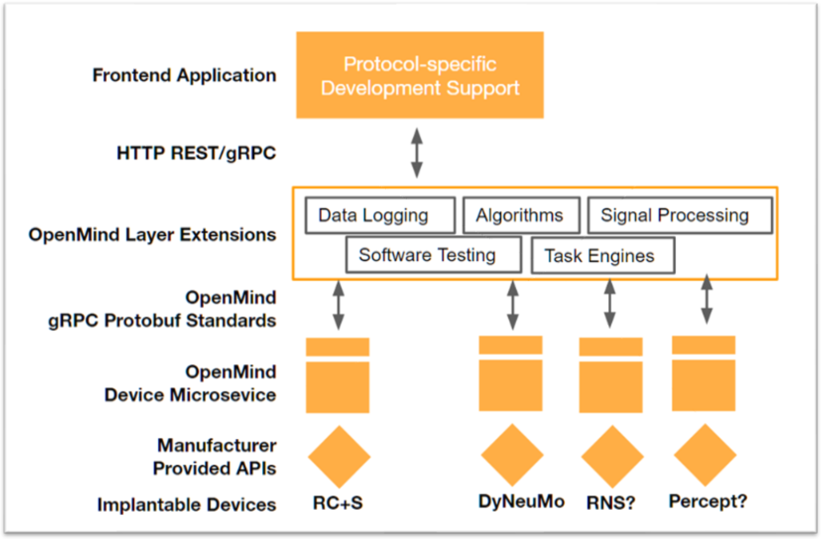
Frontend Application
Protocol-specific Development Support
HTTP REST/gRPC
OpenMind Layer Extensions
Data Logging
Algorithms
Signal Processing
Software Testing
Task Engines
OpenMind gRPC Protobuf Standards
OpenMind Device Microsevice
Manufacturer Provided APIs
Implantable Devices
RC+S
DyNeuMo
RNS?
Percept?

Figure 1 | OMNI Ecosystem demonstrating gRPC microservice placement during expected use.

To mitigate the barriers caused by INS hardware fragmentation, the OMNI Summit Microservice is being developed, an INS hardware-agnostic software framework to accelerate research in the electrical neuromodulation space. The framework relies on the gRPC, an industry-developed remote procedure call framework that facilitates software interoperability via networked application programming interfaces (APIs) [11]. gRPC uses the Protobuf interface description language (IDL) to define services, endpoints, and messages used across interfacing software. Protobuf comes with a variety of tools to generate both client and server code for 10 supported programming languages (with more languages supported through the open-source community). Further, by developing OMNI under a design control process and providing this common interoperability framework, independent research sites can efficiently bootstrap their own research protocols in whichever programming language they are comfortable with. OMNI provides an architecture for future investigational device manufacturers to use in new systems, as it provides a robust method for allowing system interoperability without requiring the development of SDKs in varying languages.

The OMNI Summit Microservice is not medical device software and is to be considered Software of Unknown Provenance (SOUP) by the medical device software incorporating it.

References:

[1] P. Khanna et al., “Enabling closed-loop neurostimulation research with downloadable firmware upgrades,” in 2015 IEEE Biomedical Circuits and Systems Conference (BioCAS), Oct. 2015, pp. 1–6.

[2] S. Stanslaski et al., “A Chronically Implantable Neural Coprocessor for Investigating the Treatment of Neurological Disorders,” IEEE Trans. Biomed. Circuits Syst., vol. 12, no. 6, pp. 1230–1245, Dec. 2018.

[3] M. Zamora, R. Toth, J. Ottaway, T. Gillbe, and S. Martin, “DyNeuMo Mk-1: A Fully-Implantable, Motion-Adaptive Neurostimulator with Configurable Response Algorithms,” bioRxiv, 2020.

[4] “Complete-System - CorTec.” https://www.cortec-neuro.com/solutions/complete-system (accessed Nov. 12, 2020).

[5] N. C. Swann and C. de Hemptinne, “Adaptive deep brain stimulation for Parkinson’s disease using motor cortex sensing,” Journal of Neural Engineering, 2018.

[6] A. Velisar et al., “Dual threshold neural closed loop deep brain stimulation in Parkinson disease patients,” Brain Stimul., vol. 12, no. 4, pp. 868–876, Jul. 2019.

[7] J. A. Herron, M. C. Thompson, and T. Brown, “Cortical brain–computer interface for closed-loop deep brain stimulation,” IEEE Trans. Neural Syst. Rehabil. Eng., 2017.

[8] V. Kremen et al., “Integrating Brain Implants With Local and Distributed Computing Devices: A Next Generation Epilepsy Management System,” IEEE J Transl Eng Health Med, vol. 6, p. 2500112, Sep. 2018.

[9] N. R. Provenza et al., “The Case for Adaptive Neuromodulation to Treat Severe Intractable Mental Disorders,” Front. Neurosci., vol. 13, p. 152, Feb. 2019.

[10] D. A. Borton, H. E. Dawes, G. A. Worrell, P. A. Starr, and T. J. Denison, “Developing Collaborative Platforms to Advance Neurotechnology and Its Translation,” Neuron, vol. 108, no. 2, pp. 286–301, Oct. 2020.

[11] “gRPC.” https://grpc.io/ (accessed Nov. 12, 2020).

4 Project Objectives

The objective of the OMNI Summit Microservice project plan is to develop a gRPC enabled microservice for the Medtronic Summit System by the end of 2022 which will allow research software engineers to develop Summit applications in any gRPC enabled language while also providing robust session management.

5 Organization, Responsibilities, and Management

5.1 Project Team

Role	Responsibility	Names
Project Leads	Heads of the project, deadline management, milestone and sprint definition. Responsible for archiving of deliverables.	David Borton, Jeffrey Herron
Community Liaison (Project Sponsor)	Open Mind Consortium community liaison. Meeting coordination and management.	Heather Dawes
Lead Developer	Primary system architect	Bradford Roarr, Jeffrey Herron
Software Developers	Developing code, tests, and corresponding documentation.	Jeffrey Herron, Priyanka Miranda, Bradford Roarr, Ivo Su, Gary Burnett

Quality Assurance	Process governor, participate in reviews as quality approver.	Fawad Jamshed, Tim Denison
Project Advisers	Advising project leads on quality, process, and best practices	Kent Leyde, Tim Denison, Jeanne Atkinson

5.2 Project Management

5.2.1 Use of Deliverables

Lists of certain documents associated with the successful completion of key activities will be identified for each project phase. These documents are referred to as “deliverables” and are to be used to provide objective evidence that the project has been executed in conformance with this project plan.

5.2.2 Design History File

The Project Leads will establish and maintain a Design History File (DHF) used to store the documents and deliverables necessary to demonstrate the project was conducted in conformance with the Project Plan. This DHF will be stored in a dedicated Google Drive folder with signed PDFs of current revisions. Old versions of signed PDFs will be maintained in a “history” subfolder.

5.2.3 Project Meetings

The Project Leads will hold a Project Meeting on a nominal weekly basis. The Project Meeting will be used to review project objectives, communicate project status, and manage project activities.

5.2.4 Verification Planning

Design (software) Verification Plans will be developed to control the implementation and execution of design verification activities.

5.2.5 Design Reviews

Design Reviews will be used to provide design oversight and feedback. Design Reviews will be organized in a hierarchical fashion. Detailed design subjects will be covered in Technical Design Reviews. Cross-Functional Design Reviews will be used to assess the results, coverage, and adequacy of the Technical Design Reviews. Cross-Functional Design Reviews require cross-functional representation and an independent observer that does not have direct design responsibility.

5.2.6 Project Phases and Phase Reviews

Project phases will be used to sub-divide the project into logical, manageable intervals. Phase Reviews will be held at the end of each project phase. Phase Reviews will require attendance by executive management, cross-functional representation, and an independent observer that does not have direct design responsibility.

Phase Reviews will provide a control checkpoint that can be used by management to actively accept, reject, redirect, or provide guidance to the project. Phase Reviews will provide an opportunity to update and review the Project Plan. Phase Reviews will include a review of deliverable items associated with the project phase. Phase Reviews will provide an opportunity to increase project visibility and communicate project status to the organization.

5.3 Software Development

5.3.1 Software Development Plan

This project plan serves as the Software Development Plan for the OMNI Summit Microservice V1. Software requirements for the OMNI Summit Microservice V1 will be described in associated Software Requirements Specification documents.

5.3.2 Software Requirement Specification Document

This project uses Software Requirement Specification (SRS) documents to capture the design input requirements for the OMNI Summit Microservice. The SRS forms the basis resulting software design and informs the development of the software verification plans.

5.3.3 GitHub

This project uses GitHub to maintain a versioned history of the source code. This project leverages GitHub features including continuous integration builds, automated testing, public issue tracker for users to report bugs, and submission of feature requests. GitHub Pull Request code review tools are used to ensure that at least two engineers have accepted proposed changes to the codebase.

5.3.4 Software Development Issue Tracking

This project uses GitHub as an internal issue tracking system, where developers of the OMNI Summit Microservice will track existing bugs, feature requests, and enhancements as part of the design, development, and testing process.

5.3.5 Source Control

This project uses GitHub hosted git repositories for source control. New features are developed on small “feature” branches. When features are complete a pull request is created and additional engineers review the code to ensure quality before accepting the changes. The feature is then merged back into the main branch.

Commit messages follow the conventional commit specification[1] (i.e. “feat: add new endpoint” where “feat” is the type of commit and “add new endpoint” is the message). Conventional commits facilitate auto generation of change logs and releases. On each pull request GitHub Actions (continuous integration product provided by GitHub) will build the source code, run static code analysis, and run all automated tests. If the analysis, build, or test fail the code cannot be merged into the main branch.

5.3.6 Building a Release

This project will create new releases with every commit to the main branch. Releases are automated through the use of GitHub Actions, the continuous integration system provided by GitHub. When code is committed to the main branch, GitHub actions will automatically build the source code, run static code analysis, run any available automated tests, update autogenerated documentation, tag the release in git, create a release page in GitHub and upload available build artifacts to the release page. Releases follow semantic versioning. The version of a build is automatically calculated based on the commit. Breaking changes (i.e. a change that removes backwards compatibility) will increment the major version, features will increment the minor version, and fixes will increment the patch version.

5.3.7 Maintenance Issue Tracking

This project uses GitHub as an external issue tracking system to give users a means of submitting bug reports and feature requests.

6 Project Phases and Deliverables

The project is divided into the following phases: Planning Phase, Development Phase, and Verification Phase.

Phases are expected to overlap (activities in the next phase may start before the current phase is closed) but will progress in the general sequence documented here.

[1] https://www.conventionalcommits.org/en/v1.0.0/

6.1 Planning Phase

The objectives of the Planning Phase are to formally define the intended use of the system and then create a valid set of requirements documents (design inputs) that describe a system that, when built, will meet developer needs for integrating OMNI as SOUP into their own projects.

Activities of this phase include:

- Conduct of technical feasibility work
- Creation of planning documents
- Collection and refinement of inputs from researchers engaged in Summit-related software development activities
- Creation and review of requirement specification documents (Design Input)
- Conduct Planning Phase Review

The following table identifies the deliverable items for this phase.

Item	**Description**	**Document ID**
Project Plan	This document, which is a combined both project plan and software development plan.	OMNI-SM001
Software Requirements Specification (SRS)	Requirements of the Microservice software	OMNI-SM002
SRS Technical Review	Meeting minutes documenting review of completed SRS document.	OMNI-SM003
Planning Phase Review	Meeting minutes documenting completion of Project Plan, SRS, and SRS Technical Review	OMNI-SM004

6.2 Development Phase

The objectives of the Development Phase are to design the software and to create and test prototype implementations of the software.

Activities of the phase include:

- Design and build of prototype software
- Conduct Technical Design Reviews
- Conduct Final Design Review
- Conduct Engineering Confidence Testing
- Design/Software Verification Planning
- Conduct Development Phase Review

The following tables identify the deliverable items for this phase.

Item	**Description**	**Document ID**
Software Verification Plan	Preliminary verification method for each requirement (one sentence). Define the number of testing protocols and high-level strategy. The plan includes unit tests and integration test strategies.	TBD
Software Design Document (SDD)	Detail the software architecture, and structure of components (modules or units). Describe use-cases and overall structure.	TBD
Interface Control Document (ICD)	Public interfaces and method calls definition. Will utilize Doxygen output (or equivalent) for public interfaces.	TBD
SDD Technical Review	Meeting minutes documenting review of completed SDD.	TBD
Release Candidates of developed software.	Includes software deliverables: ● ***OMNI Microservices Source Code, Summit System 1.X***	TBD

Item	Description	Document ID
	- ***OMNI Microservices Protobuf file, Summit System 1.x*** - ***OMNI Microservices Reference Software, Summit System, Python 1.X*** - ***OMNI Microservices Reference Application, Summit System, Javascript 1.X***	
Development Phase Review	Meeting minutes documenting completion of the development phase and associated deliverables.	TBD

6.3 Verification Phase

The objectives of the Verification Phase are to verify that the software meets specified requirements.

Activities of the phase include:

- Design and construction of test equipment and test processes
- Software Verification Testing and Test Protocol Development
- Execute verification protocols and summarize results in verification reports
- Conduct Verification Phase Review

The following table identifies the deliverable items for this phase.

Item	Description	Document ID
Test Protocols	Describes the testing step-by-step for each requirement specified in the SRS.	TBD
Test Report(s)	Report(s) summarizing the results of executing protocol(s). Includes or	TBD

	references the generated evidence of all tests documenting success or failure to meet software requirements.	
Known Anomalies	List of all known bugs and anomalies and their impact on software operation.	TBD
Release Notes	List of changes since prior release.	TBD
User Manual	Detailed documentation illustrating the use of the microservice.	TBD
Verification Phase Review	Meeting minutes documenting completion of the verification phase and the associated deliverables.	TBD

7 Product and Service Deliverable Items

The following table lists the deliverables for end-users that will be developed as part of OMNI Summit Microservice V1.

Item	Description	Notes
OMNI Microservices Source, Summit System	Source code for microservice and associated Unit Tests.	Accessed via public GitHub.
OMNI Microservices Executable, Summit System	Windows MSI Installer	MSI Installer asks for Summit DLL locations.
OMNI Microservices Protobuf file, Summit System	gPRC protobuf file that provides microservices interface between users application and a Summit System.	Utilizing the GRPC.io supported tools utilizing the provided protobuf file.
OMNI Microservices Reference Code, Summit System, Python	Training code showing basic functionality of microservices in Python: ● Connect and Disconnect ● Getting the battery levels ● Doing a lead integrity test ● Configuring sensing ● Starting a data stream	Simple .py files that show how to use the SDK that runs in the command line.
OMNI Microservices Reference Application, Summit System, Javascript	Utilizes Electron to provide a Javascript based desktop application. Functionality: ● Connect to two Summit RC+S devices ● Monitor battery levels ● Perform lead integrity test ● Configure sensing ● Starts the streaming of time domain data & power.	Integrated application that will run out of the box (with electron setup) for users to perform functionality without writing their own code.

OMNI Summit Microservices Documentation	Support documents including: ● User Manual (includes code snippets) ● Software requirements specification (SRS) ● Interface Control Document (ICD) ● Software Design Document (SDD) ● Known Anomalies LIst (KAL) ● Release Notes ● Testing Plan and Results Document ● Soup Analysis for ***OMNI Microservices Reference Application, Summit System, Javascript 1.X***	User manual includes creating new gRPC projects from scratch. Release Notes will be generated by transitioning GitHub commits to follow the Conventional Commits standard.
OMNI Microservices SDK, Summit System	A distributable software package that Includes the following: ● ***OMNI Microservices Executable, Summit System 1.X*** ● ***OMNI Microservices Protobuf file, Summit System 1.x*** ● ***OMNI Microservices Reference Software, Summit System, Python 1.X*** ● ***OMNI Microservices Reference Application, Summit System, Javascript 1.X*** ● ***OMNI Summit Microservices Documentation***	A .zip file that contains separate folders for deliverables and an MSI Installer.

8 Reference Information

8.1 Definitions

Term	Definition
build artifact	The results of compilation, any libraries or executables required to run the software
continuous integration	The practice of merging development code into a centralized branch generally facilitated by automated tooling
endpoint	Remote procedure call that allows interfacing software to access some feature
GitHub Actions	A framework for creating jobs to run on different events within the continuous integration cycle (e.g. pull-request, push, merge, release, etc.)
GitHub Issues	A tool tied to a repository within GitHub used to track issues and feature requests from consumers of the software
gRPC	Remote procedure call framework originally developed by Google. See https://grpc.io for more details
service	A logical grouping of endpoints to provide access to similar functionality. Not to be confused with microservice

main branch	The centralized branch containing all of the latest code. Releases are cut from the main branch
message	The data passed to and from an endpoint
microservice	A single networked executable designed to work in tandem with other small executables
protobuf	An interface description language used to generate network client/server code as well as message serialization

9 Approvals

Approver Role	Signature and Date
Project Lead	DocuSigned by: David Borton 8A588B8FA1F246E... 1/30/2022 | 10:54:58 PM EST
Quality	DocuSigned by: Fawad Jamshed E3B1537628454D3... 2/7/2022 | 9:08:09 AM EST
Lead Developer	DocuSigned by: Bradford N Roarr 9617A5434271462... 1/31/2022 | 8:19:14 AM EST
Author	DocuSigned by: Jeffrey Herron 64503D12F815426... 2/2/2022 | 10:50:42 AM PST