

Developing Surgical Planning Platform for Long-Term Sustainability: One Decades Worth of Lessons

Dr. Rachel Sparks
School of Biomedical Engineering & Imaging Sciences
King's College London

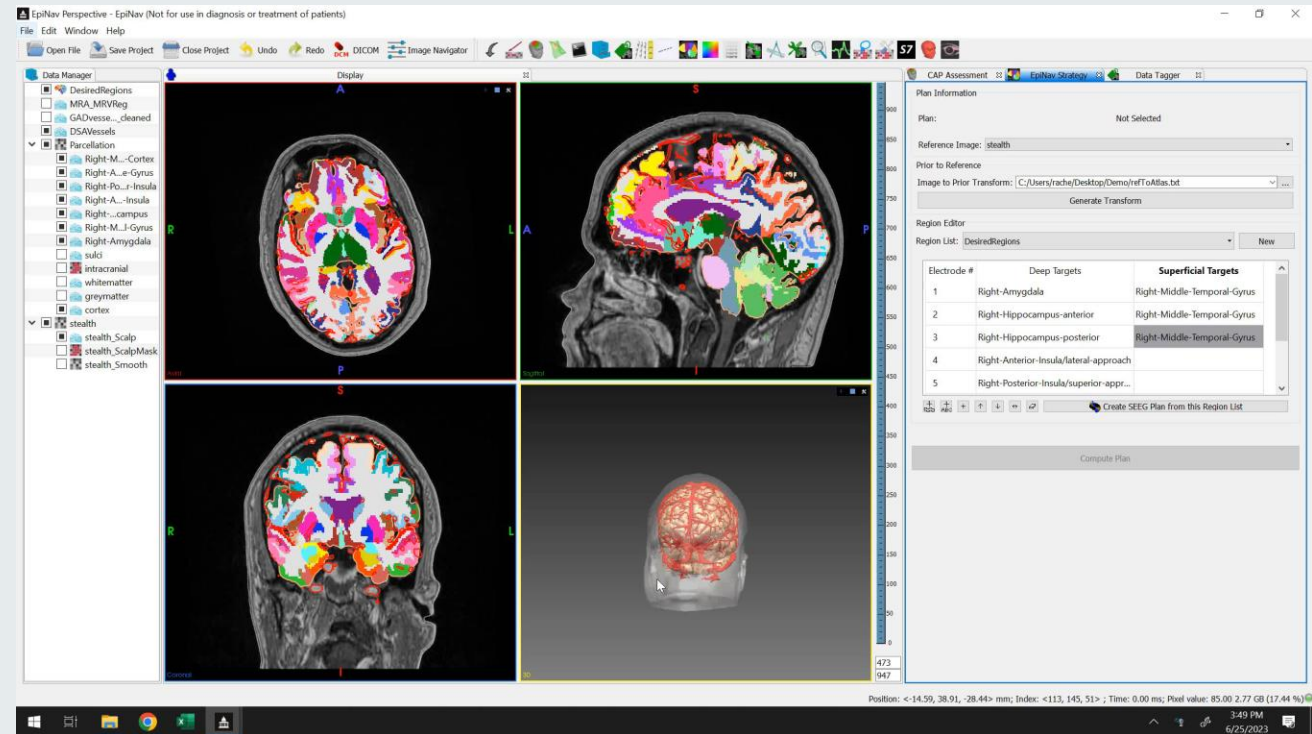
Overview of talk

Provide some of my insights into development using Open Source Software for Medical Purposes

- Projects overview: open source and not so open source contributions
- Key considerations for Open-Source Software with a medical purpose
- Open challenges

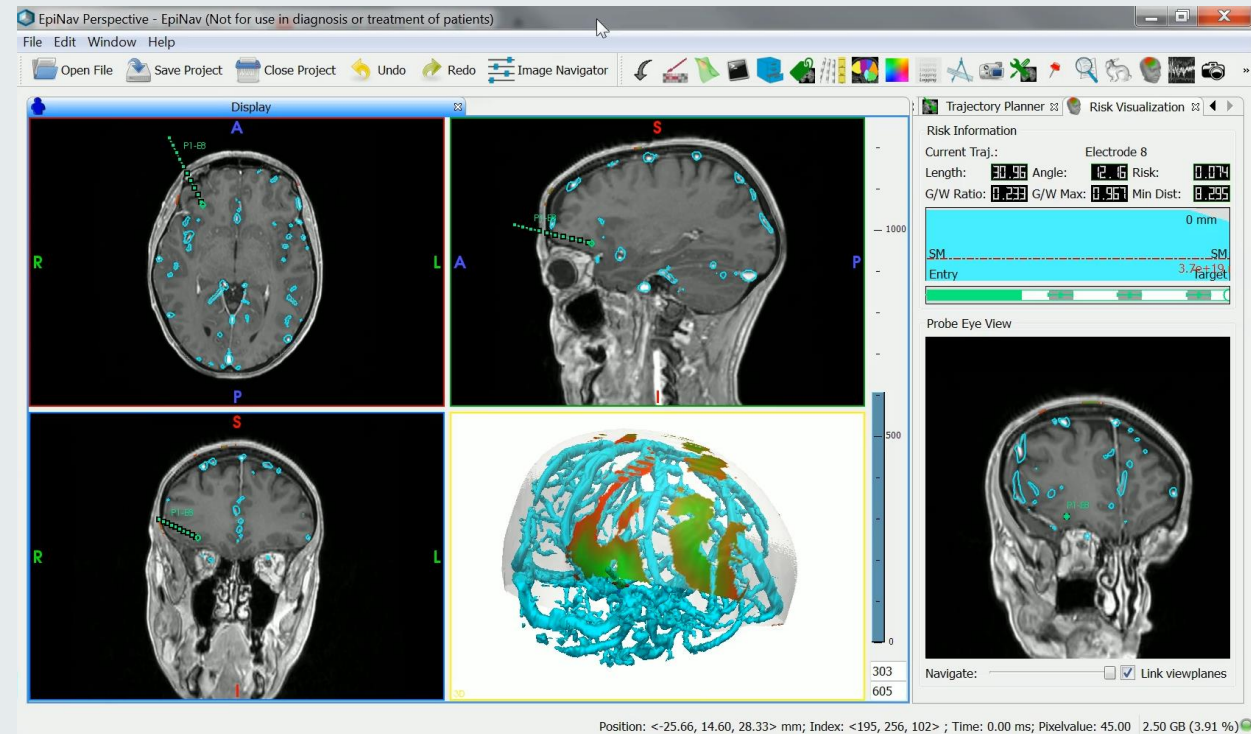
Epilepsy Navigation (EpiNav™)

- Built using MITK architecture as a standalone library with platform specific plugins
- Started in 2013: original part of the NifTK platform (with MIDAS, SmartLiverSurgery)
- This is a surgical planning software to plan stereotactic neurosurgery trajectories
 - Support interface with commercial neuronavigation systems
 - Since 2018 used in prospective clinical studies at NHNN & UMC Utrecht to plan SEEG implantations*



Epilepsy Navigation (EpiNav™)

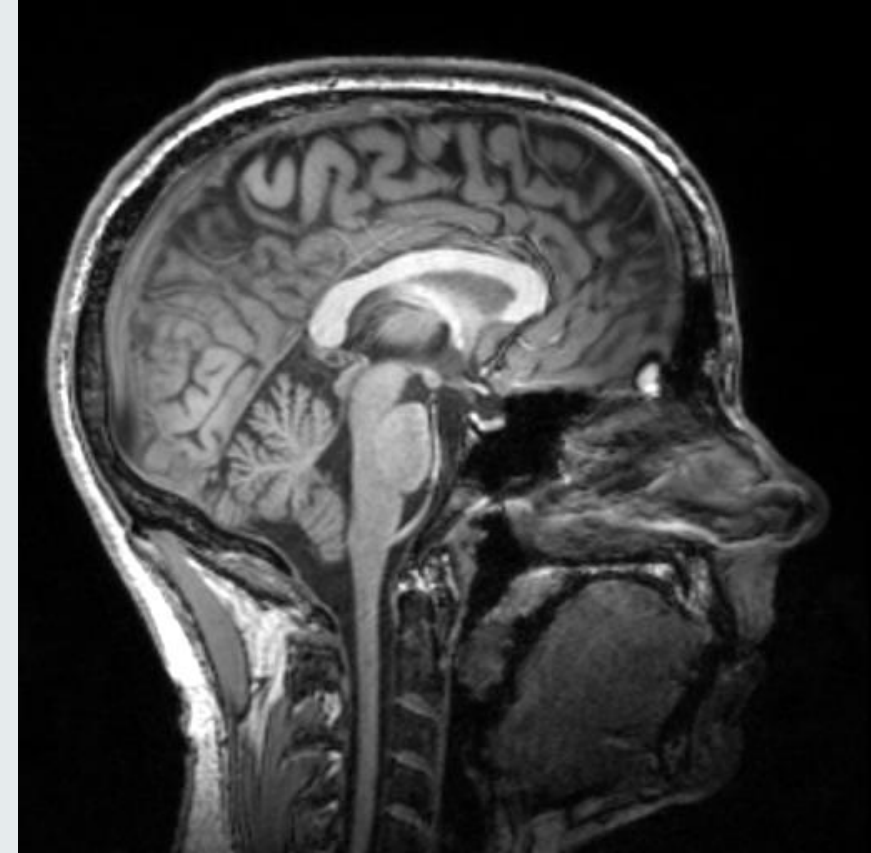
- Built using MITK architecture as a standalone library with platform specific plugins
- Started in 2013: original part of the NifTK platform (with MIDAS, SmartLiverSurgery)
- This is a surgical planning software to plan stereotactic neurosurgery trajectories
- End goal is commercialization so key components of the algorithm are closed source.



Other (shorter) projects

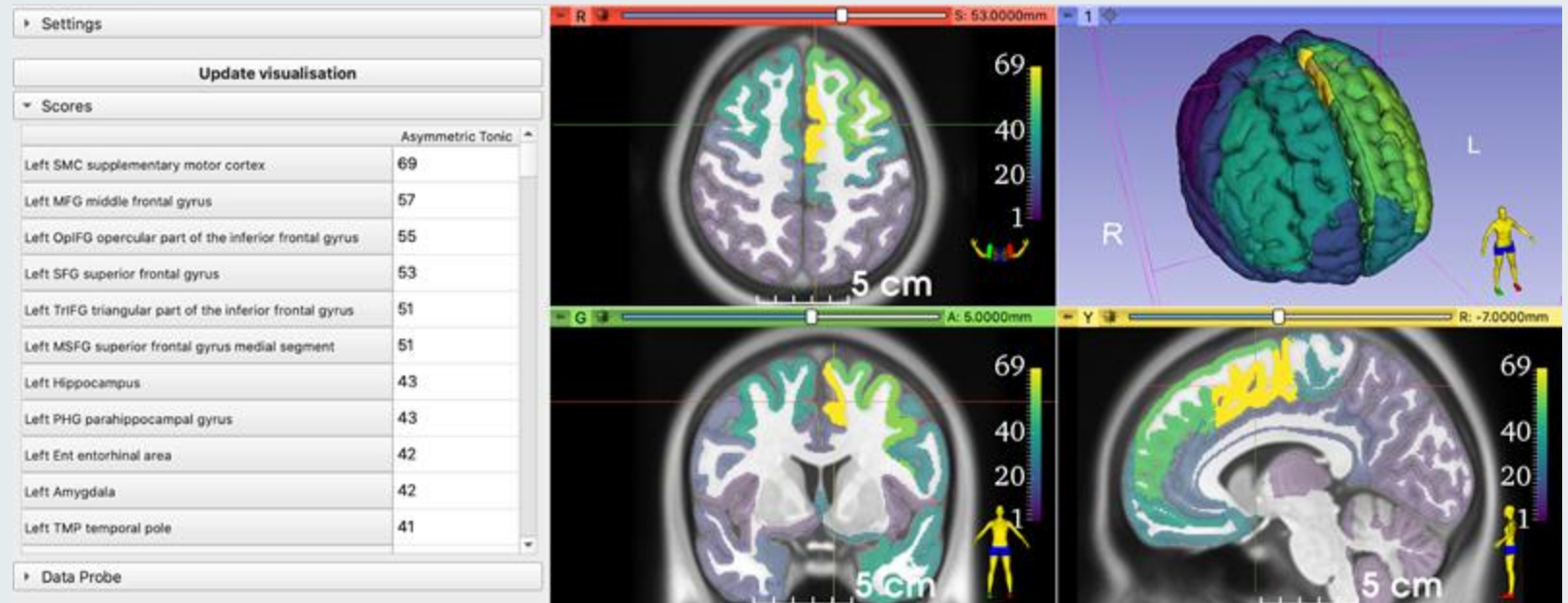
TorchIO (<https://github.com/fepegar/torchio>)

- Python package to support data loading and preprocessing for deep learning
- Started as a single person project (Dr. Fernando Perez-Garcia)
- Created due to lack of current tools
- Research tools for researchers for easy use and avoid recoding of basic data loading



Other (shorter) projects

- Semiology Visualisation Toolkit
(<https://github.com/thenineteen/Semiology-Visualisation-Tool>)
 - Slicer plugin to help query and visualize a database of epilepsy semiology associations
 - Created to allow clinicians to visualize a large amount of data and explore hypothesis
 - Research tool for clinicians to link patient symptoms to data in the literature



Other (shorter) projects

- I encourage everyone on my team to open source their code (both methods and experiments)
- **Open-source for scientific reproducibility \neq creating an open source toolkit (platform)**
 - Reproducibility: documenting what you did (parameters, optimization) and how you did it
 - Toolkits: code that is (hopefully) useful focused on reducing need to recode the same solution

Overview of talk

Provide some of my insights into development using Open Source Software for Medical Purposes

- Projects overview: open source and not so open source contributions
- **Key considerations for Open-Source Software with a medical purpose**
- Open challenges

Consideration 1: Project Goal

End goal of the project influences every other design choice

Research tool for Researchers

- Flexible, extensible code
 - Common programming language
- Open interfaces
- Well documented
- Ignore front end
- Limited ability to reach clinic

Research tool for Clinicians

- User interaction
 - Clear easy graphical user interface, accessible documentation
 - Support staff to assist in running
- Integration into clinical workflow
- Feedback from end users
- Ethics, regulatory considerations

Licensing/Commercialisation

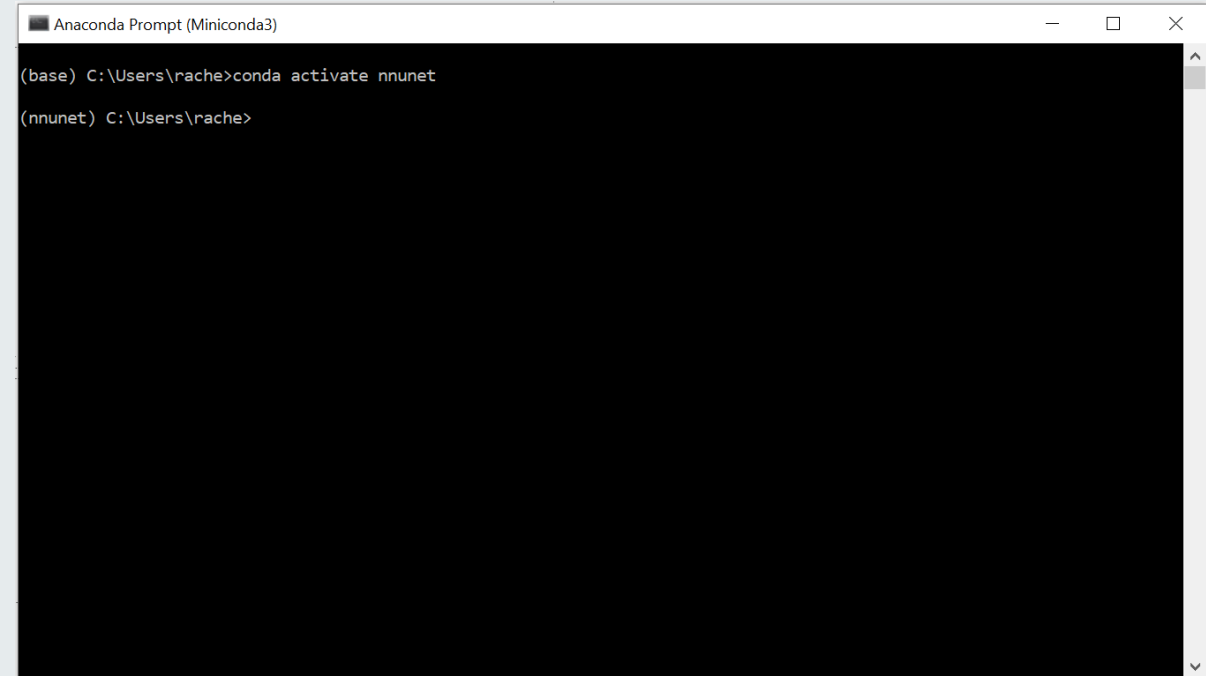
- Protect IP
- Demonstrate benefit
 - Ensure key algorithms work “well”
 - Impact on surgical workflow
- Quickest route to clinical impact
- Frontend effort likely going to waste

What is your project lifespan and how do you maintain support over it?

Consideration 2: End-Users

Who is your end user and what is their technical competency?

- Students/Researchers
 - Do you need a UI
 - What programming language will they be familiar with
 - Any barriers in terms of access to computational resources (GPU servers? Specific hardware or OS)
 - Other platforms or OSS tools you want to be compatible with



```
Anaconda Prompt (Miniconda3)

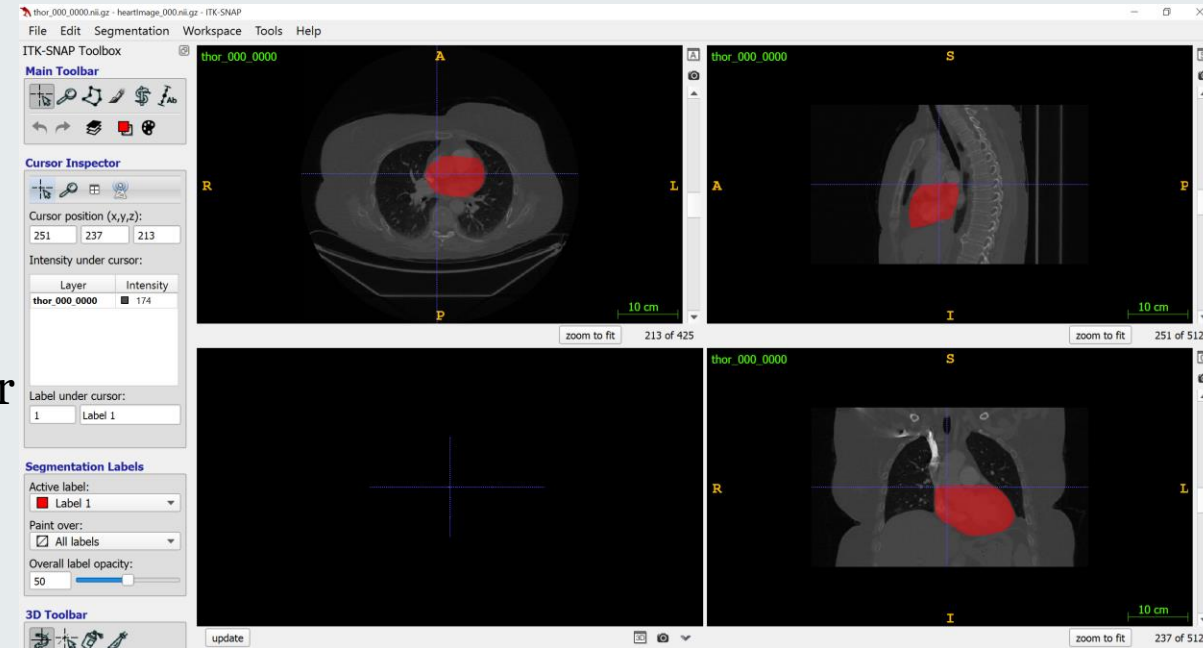
(base) C:\Users\rache>conda activate nnunet

(nnunet) C:\Users\rache>
```

Consideration 2: End-Users

Who is your end user and what is their technical competency?

- Clinicians
 - What is their computer literacy – can they use command line interfaces?
 - What sort of specialties do they have? Do they have other computer software they are familiar with
 - Mouse/key board interactions mimic PACS or another familiar software
 - Radiological versus neurological convention?
 - How busy are their routines? How much time can they spend for troubleshooting

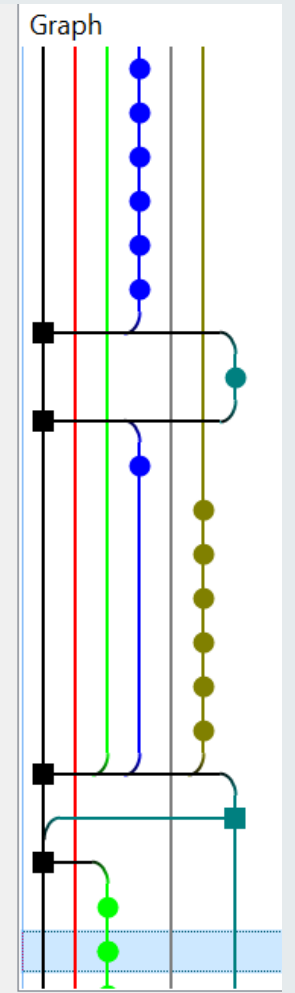


Researchers and students have a much higher tolerance for errors and unexpected behavior

Consideration 3: Sustainability

Who is your development team?

- PGRs? Post-docs? Professional developers?
 - How do you pay competitive salaries
 - Do they need training? On the language, on the platform, on specific code modules
 - What is their “job” : most people in academia are expected to have research outputs
 - Career progression & average employment period
- Team continuity
 - Who knows why (seemingly stupid) decisions were made
 - How to achieve a culture of good coding practices (documentation, bug tracking, etc.)
- Who will do the thankless work
 - Who can support dependencies upgraded, CI Runner maintenance
 - How to deal with push requests, ensuring code standards



Consideration 3: Sustainability (cont)

Who is your development team?

How are you going to fund the team?

- Most funding agencies want research or development goals
- Not just developers, need systems for continuous integration and/or nightly runners
- Hosting needs documentation, repository, bug tracker, binaries
 - Not necessary but can be helpful to maintain a presence

Consideration 3: Sustainability (cont)

Who is your development team?

How are you going to fund the team?

Community Events

- Developer meetings
- User meetings
- Hackathons or project weeks to support multi-institutional work

Do you need your own platform?

Many great existing open source platforms already exist:



Do you need your own platform?

Many great existing open source platforms already exist:

Do you have a unique selling point?

- Unmet need in the community that you will address (i.e. will you have users)
- There is a high financial and development burden to create your own platform
- Maintenance once created vital to ensure continued users

Do you need your own platform?

Many great existing open source platforms already exist:

Do you have a unique selling point?

If you decide to go to with an existing platform be a good community member:

- Report bugs. Better yet fix bugs and make push requests
- Engage with the community (hackaton/project weeks, post to forums, attending development meetings)

Overview of talk

Provide some of my insights into development using Open Source Software for Medical Purposes

- Projects overview: open source and not so open source contributions
- Key considerations for Open-Source Software with a medical purpose
- **Open challenges**

How to Balancing Research & Code

Ultimately software tends to be supported by research grants that have research (i.e. publication) goals

- Silo developers and researchers
 - Often PGRs don't have strong software development skills
 - Software developer career progression can stall in academia and salary levels are well below industry standards
 - Guarantee research doesn't get postponed for bug fixes
- Mix development and research roles
 - Need hands on training to get development skills up to speed
 - Slower development times, easy for thankless tasks to be left undone
 - Ensures research is fit for purpose and gets incorporated into the software

Open-Source with Regulatory Approvals?

If the end users are clinicians regulatory approval for medical devices required

- Quality management system in place
- Well defined system to track software quality, report and fix bugs
- Typically a tightly controlled environment with well defined roles and hierarchy
 - Approval for code changes
 - Test coverage must be very high and well maintained

How to best support open-source research while ensuring improvements make it to the patients

- Almost always to have widespread impact on the clinic you need to involve a commercial entity
- Qt, and others have addressed this issue by having both free and commercial versions -> the code base has an appropriate license
- Create spin-out using know how from the open-source software
- Protect the key IP by withholding portions of your source code

Concluding Remarks

For open source project sustainability

- Define the project goal
- Understanding the ecosystem
 - What tools (open source and commercial) exist and are in use?
 - Who is a user and who is a contributor?
 - Other programming languages or platforms with similar tools?
- Funding strategy to support appropriate staff and resources
- Project management well defined to recruit and maintain highly skilled staff