BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Remote Access to Clinical Data

Lo-fi prototyping and the spiral lifecycle

Andrew Schneider (andrew.schneider@bjss.co.uk)
Technical Architect

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

The Problem

- Vision statement:
  - “We would like to provide distributed access to a patient’s clinical […] we’d like it now.”
- Constraints:
  - Cost.
  - Time to market.
  - Privacy.
  - Politics.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

The Problem

- How to migrate from the above "requirements" to a working system.
  - How do we gather requirements?
  - How do we manage change in requirements?
  - How do we deliver quickly?

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Selecting a Lifecycle

- A lot of projects have an emergent lifecycle.
- A conscious choice improves a project's chances of success.
- We rejected:
  - XP.
  - Staged.
  - Waterfall.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Selecting a Lifecycle

- We chose the spiral model:
  - Applicable to projects without clear requirements.
  - Focused on reducing risk as project continues.
  - Provides a framework for managing and incorporating change.
- We were careful to:
  - Define success criteria for each iteration.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Iterations

- Initial prototyping (3 weeks).
- Demonstrable prototype (6 weeks).
- Core functionality (8 weeks).
- First candidate release.
- Final revision.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Additions

- We took the “test infected” model from XP.
- Made developers more comfortable with change.
- Found the initial “loss of productivity” hard.
- Enabled us to use the architectural prototype as a base for production.

BJSS
INNOVATIVE SOFTWARE DESIGN & DEVELOPMENT

Initial Prototyping – Previous Experiences

- Domain experts find it difficult to work with abstractions.
- Domain experts never agree.
- Domain experts can have trouble prioritising.
- Domain experts aren't usually dedicated to the project. Need to maximise use of their time.
- No common language.
- End result: Lack of co-operation.

BJSS
INNOVATIVE SOFTWARE DESIGN & DEVELOPMENT

Initial Prototyping – Working Prototypes

- Working prototypes are often a combination of:
  - "GUI builder"
  - Bespoke development.
- Takes a developer to change the prototypes.
- Prototypes give false expectations.
- The computer can be a distraction.

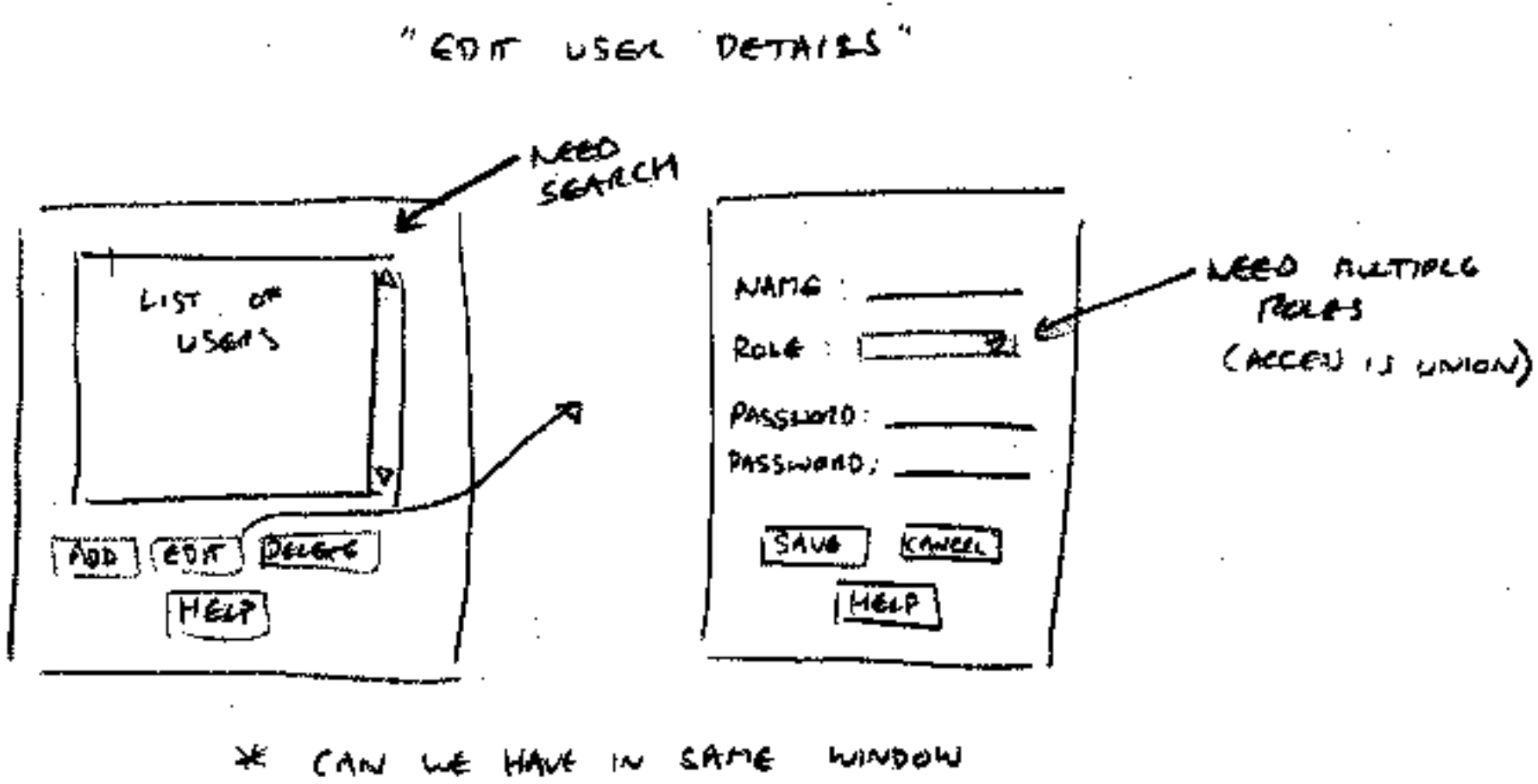
BJSS
INNOVATIVE SOFTWARE DESIGN & DEVELOPMENT

Initial Prototyping – Lo Fi Prototyping

- Reverted to pencil and paper.
- Structured diagrams for common use cases.
- Dynamic behaviour added with arrows and story boards.
- We could change the prototype during a meeting.
- We built an object model at the same time.

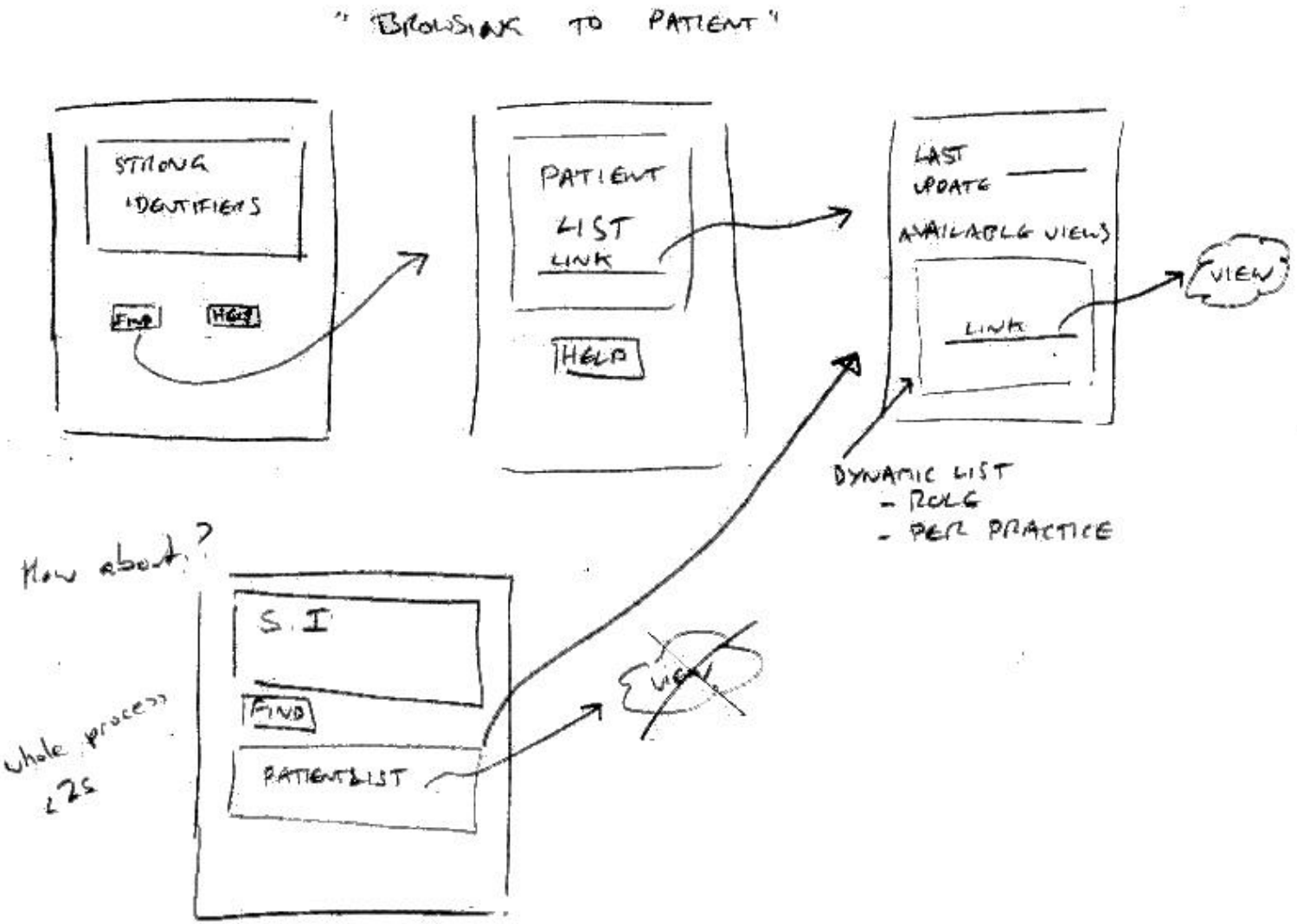
BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Example: “Edit User Details”



BJSS
INNOVATIVE SOFTWARE DESIGN & DEVELOPMENT

Example: "Browse to Patient"



BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Initial Prototyping – What We Didn't Do

- Follow the whole lo-fi prototyping process.
- No formal evaluations by users.
- No record keeping.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Initial Prototyping – Results

- Paper didn't get in the way.
- Interaction was more dynamic.
- Domain experts could draw on the diagrams.
- E-mailed scanned images.
- User interface discussions raised other issues.
- Prompted comments such as "try such and such idea… Mr XXXX has a prototype of that user interface element."

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Initial Prototyping – Results

- 2-3 iterations of a use case in a 30 minute meeting.
- Non-functional constraints were added.
- Greater co-operation than we had previously experienced.
- No software, less pressure to ship early.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Initial Prototyping – Scaling

- Scaling improved by:
  - Partitioning use-cases.
  - Partitioning top level user interface.
  - Sharing a common object model.
- Scaling problems:
  - Difficult to track fine grain requirements.
  - Little support for common interface components.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Demonstrable Prototype

- Static web site.
  - Allowed people to see the mapping between the drawings and the computer.
  - Needed little change as most of the issues already ironed out.
- Architectural prototype.
  - Using JINI was a risk.
  - Security was a risk.
  - Ensured we could meet performance requirements.
  - Allowed us to measure development “velocity”.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

Core Functionality

- Domain experts prioritised.
- Estimated each diagram.
- Factored in velocity from prototype.
- Chose diagrams to fit into development time.

BJSS
INNOVATIVE SOFTWARE
DESIGN & DEVELOPMENT

First Candidate Release & Final Revision

- At this point the project is heavily committed.
- Only very minor changes fed in (or accepted).
- Delivery was within 10% of predictions.