

# Dispersed Development

OT2004 Case Study

John Daniels and Paul Dyson

fastnloose

$e^x$

# What is dispersed development?

Different flavours of the dispersed problem:

- Multiple teams
  - Large undertaking with multiple teams, each team in one office but teams in various locations around the world. Frequently encountered in large companies.
- Satellite workers
  - Most of the team in one office but a few working remotely.
- Dispersed team
  - What we are considering here. Every team member working remotely.

# Why do it?

- Costs
  - workers based at home can be much cheaper
- Quality
  - allows access to good people who don't want to commute / work away
- Constrained by situation
  - multi-site projects with distributed team members
- Individual constraints
  - personal commitments such as doing childcare at home

## Sources - JD

- JD: Two FNL projects considered here
  - Both part of a larger project
  - Project 1 (1)
    - Specialised workflow engine: Two people, fixed price, fairly self-contained, 5 months elapsed, J2EE, no UI
  - Project 2 (2)
    - Information management: Four/five people, T&M, a lot of inter-dependence on other (non-FNL) teams, 6 months elapsed, J2EE, no UI

## Sources - PD

- PD: Two e2x projects considered here
  - Project 1 (1)
    - Integration of web services provision to Internet platform: Four/five people, fixed price, 3 months elapsed, Java, no UI
  - Project 2 (2)
    - Development of e2x product: Four/five people, 4 months and counting, Java, multiple interfaces, migrated from office environment
- **All projects followed agile/XP process**

# Topics

- The working environment
- Interacting with the customer
- Project process
- Designing the system
- Team organisation
- Programming
- Testing
- Delivering the system
- General advice

# The Working Environment (part 1)

- Minimum level of technology support essential:
  - Full broadband
    - 512k ADSL/Cable minimum (beware upload speeds!)
    - ISDN too slow
    - Satellite latency too high
  - Phones plus conference call facility
    - Can use Voice over IP for one-to-one communication
  - Powerful machines
    - But must limit desktop space for efficient sharing
    - Needed to install complete platform (Oracle, app server, etc) in each location
  - Wiki
  - Version control system
  - NetMeeting for collaboration
    - VNC or RemoteAdministrator for control
  - Instant Messaging for chat & to see who's around
  - Third-party hosted server?
  - Security?
    - Firewall
    - IPSec encrypted VPN (all use same model of router)
    - encrypted email (PGP)
    - if appropriate, encrypt disks (PGP)

## The Working Environment (part 2)

- Tool support currently very poor
- Not easy supporting many and varied desktops! Lots of configuration slowdowns - everyone has to be their own system manager

# Interacting with the customer (part 1)

- General
  - Initial f2f meeting essential, followed by regular f2f reviews
  - Can't support 'normal' model of XP customer
    - More expectation management required because of lack of shared experience
    - More formal review periods (make sure the customer looks at it!)
    - More formal record of what the customer asks for
    - Need to book time with the customer rather than approach them in ad-hoc manner
    - More documentation
  - Issues raised/resolved by email/phone/wiki
  - Customers had no access to VPN or servers

# Interacting with the customer (part 2)

- FNL
  - Customer had already done some prototyping (1)
  - Customer had already done extensive business analysis and produced requirements documents (2)
  - Jointly produced feature list (1&2)
  - We had access to their intranet and repository (2)
  - Customer very happy with dispersed development - no office space! (1&2)
- e2x
  - All requirements captured by e2x people (1&2)
  - Prioritisation carried out jointly (1&2)
  - No off-site access to customer servers (1)
  - Customer very suspicious of dispersed development (1)

# Project process: general

- Evolutionary approach
- Short (one- to three-week) iterations essential (even more!)
- Daily (or more often) “stand up” meetings
- Features vs tasks
  - customer defined features, we defined tasks
- Prioritised task list with estimates
  - wiki or shared excel spreadsheet (or word doc)

# Project process: FNL

- Project leader updated spreadsheet during daily meeting
- We didn't track estimates
- FNL P1
  - Captured requirements (for phase) and estimated onsite (1 day)
  - 4 two week iterations, with delivery at each iteration
  - Customer handled integration with little FNL input required
  - Very similar for phase 2
- FNL P2
  - Many onsite sessions to clarify requirements
  - 2 to 3 week iterations, with delivery at each iteration
  - 1 or 2 days onsite to assist with integration at each major delivery
  - 1 week+ onsite after final delivery to really integrate

# Project process: e2x

- Used Wiki forms – visible to customer as well as everyone else
  - Use as virtual wall
  - Hard to enforce, discipline required
- e2x P1
  - Captured requirements onsite and estimated (three days)
  - 5 one-week iterations of dispersed development
    - Weekly detailed planning
  - On-site delivery with integration and review leading to new requirements and estimates (four days)
  - 4 further one-week iterations
  - On-site delivery with integration, review and handover
  - Go-live
- e2x P2
  - Rolling ‘roadmap’ of requirements.
    - Customer prioritises at this level
  - Weekly iterations
    - Confirm detailed requirements with customer – can be very brief
    - Plan week
    - Do week’s work
    - Take new version live
    - Update roadmap

# Designing the system (part 1)

- General
  - Early and key architecture/design decisions made during face-to-face working
- FNL
  - Some use of shared whiteboard to sketch designs (mostly in UML)
  - Rose UML model for entity beans (held in CVS)
    - documented the information model
  - API-driven design approach
  - (Functional) Design documentation “after the fact”

## Designing the system (part 2)

- e2x
  - Revisit architecture/design decisions regularly in f2f
  - More important than in 'normal XP'
    - Lack of 'overheard' communication
    - Lack of coffee/lunch discussions
  - Very poor tool support for sketching ... modelling okay (use modelling tool and shared desktop)

# Team organisation

- FNL
  - Mixture of true pair working and “buddying” - Buddies work alone but confer frequently (typically every 30 mins - 1 hr)
  - When pairing, take next task and
    - agree in the pair how it is to be done
    - if complex or there is uncertainty
      - work as a pair until complexity or uncertainty removed
    - then if work remains to complete task
      - split all or part of the task between the pair and become buddies
      - one common split: tests vs. implementation
- e2x
  - All tasks paired - ‘Lone ranger’ job queue for the odd singleton

# Programming: technology

- Eclipse
- CVS
  - Eclipse and Tortoise clients
  - Server running on spare PC (cf. server hosting)
  - e2x: later switched to Cyclex
- Pair programming using Net Meeting over broadband
  - slight delay for non-local user, acceptable if not driving (okay for the odd click)
  - re-sync to version control to switch driver
  - Mostly problem-free!
- Varying degree of continuous integration

# Programming: task process

- Process for programming tasks:
  - Sketch the interfaces
  - Code the tests
  - Run the new tests
  - Code the implementation
  - Run the new tests
  - Debug until the new tests pass
  - Run all tests and debug as necessary
  - Check-in the new code
- Test-focussed rather than 'just' test-driven
- Major refactoring was done as separate defined tasks
- Updated the user and design documentation before each software release

# Programming: FNL

- Customer's repository, no integration with Eclipse (P2)
  - easy to forget to check in new files!!!
  - couldn't use NetMeeting while connected to their repository, made pair programming tedious - keep stopping and starting
- Pair programming using Net Meeting over ADSL
  - once you have the VPN working there are no firewall / port issues
  - audio and desktop sharing (must have headsets)
- Issue: pair need to re-synch before task is complete, but don't want to make their interim work available to others

# Programming: e2x

- Who's got the monkey?
  - Use of instant messaging
- Lack of 'overheard' communication
  - Interruptions worse than when co-located
  - Regular review calls
- Lack of lunch conversations
- Latency makes regular swap-over necessary
  - Poor support in CVS
  - Supported in Cyclex!

# Testing

- General
  - Focus on functional tests ... are unit tests still important?
  - Junit
  - Tests fully automated (one click)
  - Automated testing absolutely critical
    - Best communication mechanism between pairs
- FNL
  - When pair working
    - **both** developers get all tests to pass on their machine before agreeing the task is complete

# Delivering the system (part 1)

- Physical delivery
  - Onsite delivery (e2x P1)
    - Customer security didn't allow remote access to servers
  - Remote delivery (e2x P2)
    - Fully automated delivery – can be run by anyone
- Remote delivery (FNL)
  - zip of source, release notes, user and design documentation (P1)
    - Customer rebuilt and checked our delivery, then added it to their repository (different technology)
  - direct use of customer repository (P2)

## Delivering the system (part 2)

- Integration
  - Onsite integration at end of project (e2x P1)
  - Continuous integration (e2x P2)
  - Integration at discretion of the customer (FNL P1)
  - Periodic integration (FNL P2)
    - But not complete enough!
    - Significant time required to integrate our deliver with rest of system - not always clear whose responsibility that was!

## Delivering the system (part 3)

- Handover
  - Customer techie joins integration, lots of documentation (e2x P1, FNL P2)
  - Extensive release notes (e2x P2, FNL P1)
- Feedback
  - Offline feedback (no issues with delivery) (e2x P1)
  - Feedback through wiki and f2f meetings (e2x P2)
  - Feedback through phone/email (FNL)

## General advice

- Shared understanding is the key at all stages of the work
  - e2x: Does work with people you don't know – works better with people you do
  - FNL: We only employed people we already knew!
    - minimised start-up time
    - minimised cultural and personal differences
- Some degree of f2f is necessary – particularly for customer relationship
- Establish productive environment up-front
- Establish working etiquette as you go
- Allow time to set up infrastructure (servers, routers, VPNs, etc)
- Test focus (but we don't know how to do it another way)
- Team size limit? 5+?