Distributed Component Engineering using a Decentralised, Internet-based Environment

John Grundy

University of Auckland
New Zealand

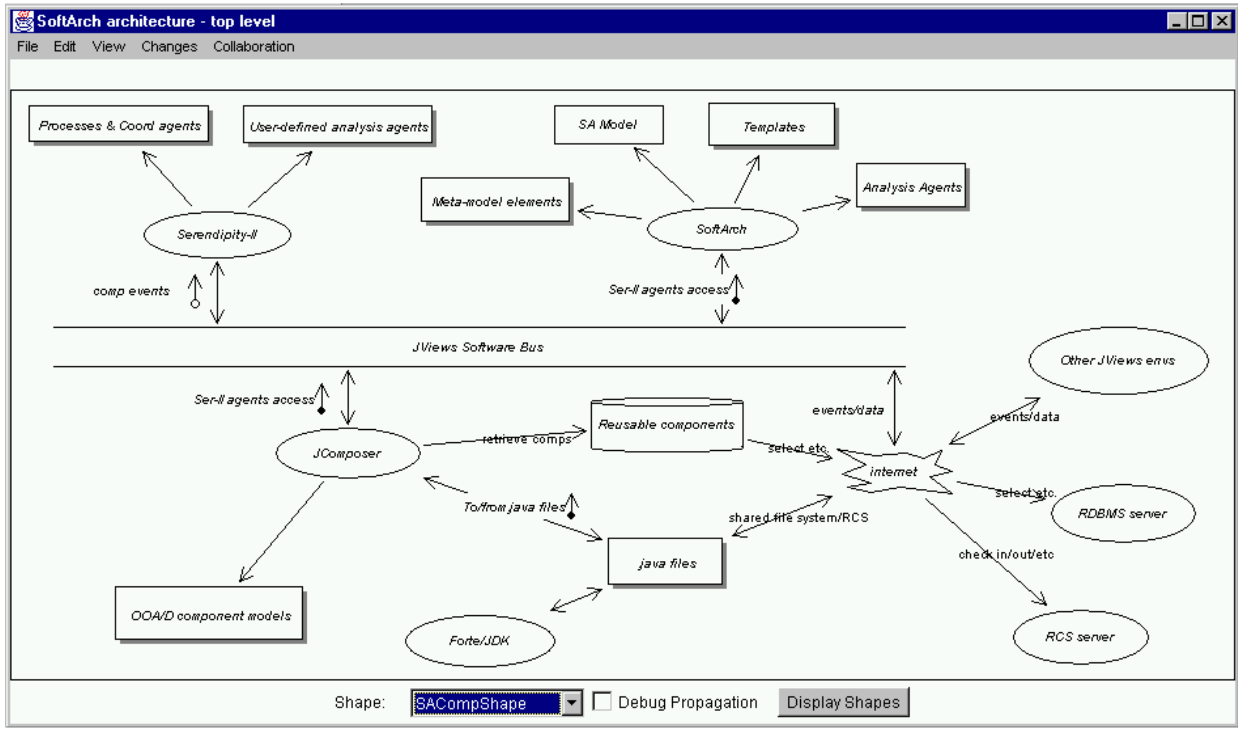
Outline

- Component Engineering
- Our component engineering environment
- Example facilities:
  - Process modelling & enactment
  - Collaborative editing
  - Distributed work co-ordination & tool integration
  - Component management
- Building such an environment (briefly!!)
- Future work/Conclusions

Component Engineering

- Idea of discrete, reusable software components
- Maximise reuse; maximise run-time reconfiguration/deployment
- Share/buy components; support diverse component interaction
- Need variety of tools to DO this: process, co-ordination, CASE, implementation, library, testing, deployment, ...

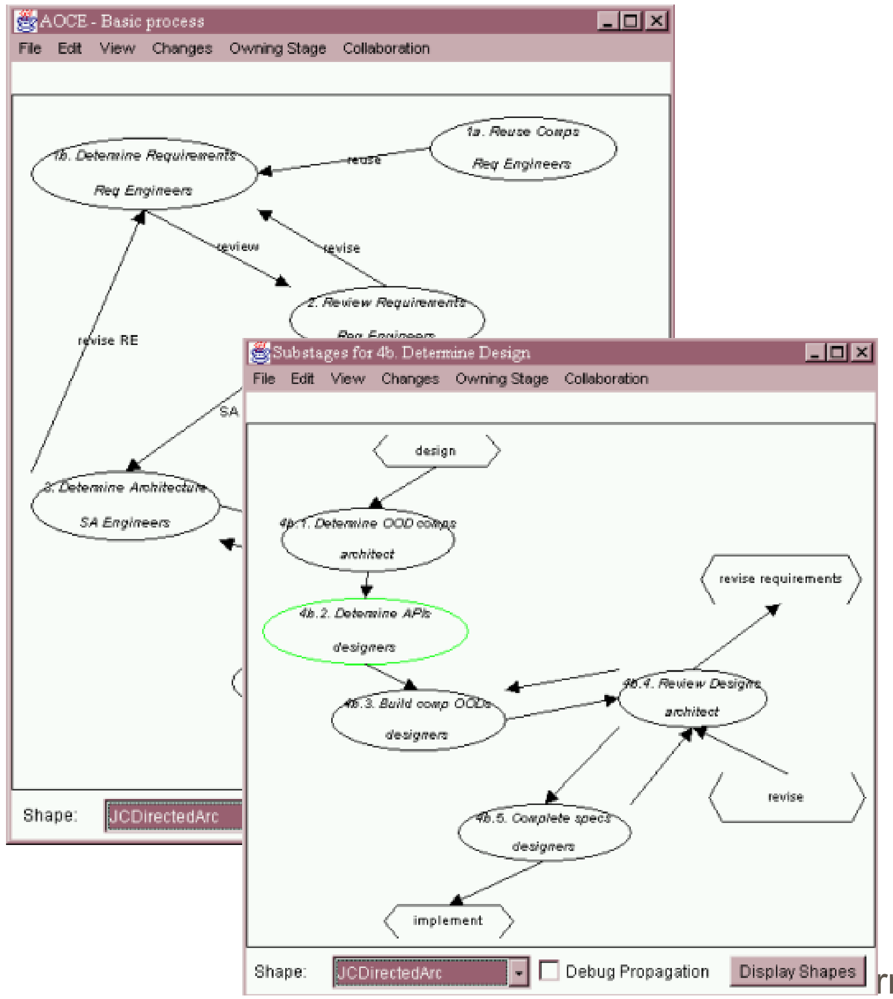
Our Component Engineering Environment...



Shape: SACompShape

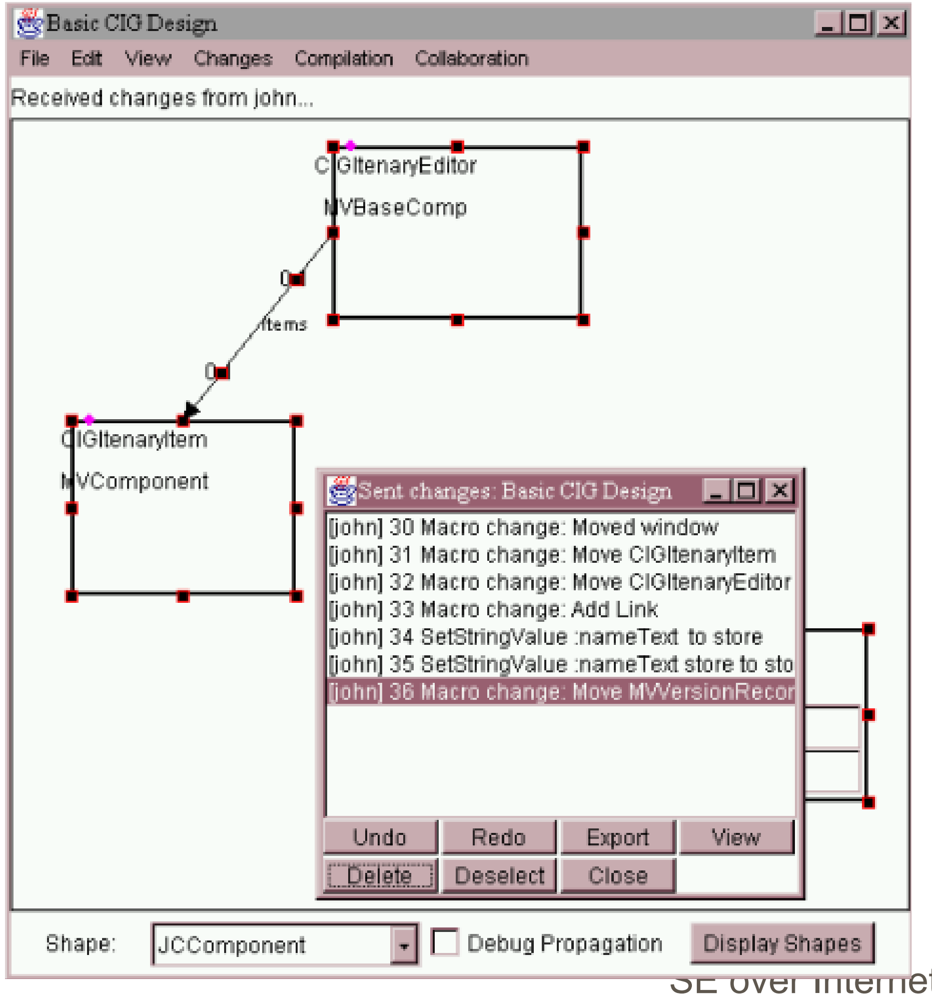
- Serendipity-II:
  - processes/agents
- SoftArch:
  - High-level component groupings
- JComposer:
  - CASE/impl.
- JVisualise:
  - debugging
- Component Library:
  - reuse
- Others (DB, RCS, Forte)

Process Management



- Via Serendipity-II process management tool
- Model processes, roles, resources, instantiate work plans
- Enact processes
- Collaborative modelling
- Process/plan templates
- Task automation/tool integration agents
- Co-ordinate use of other tools…

Collaborative Editing

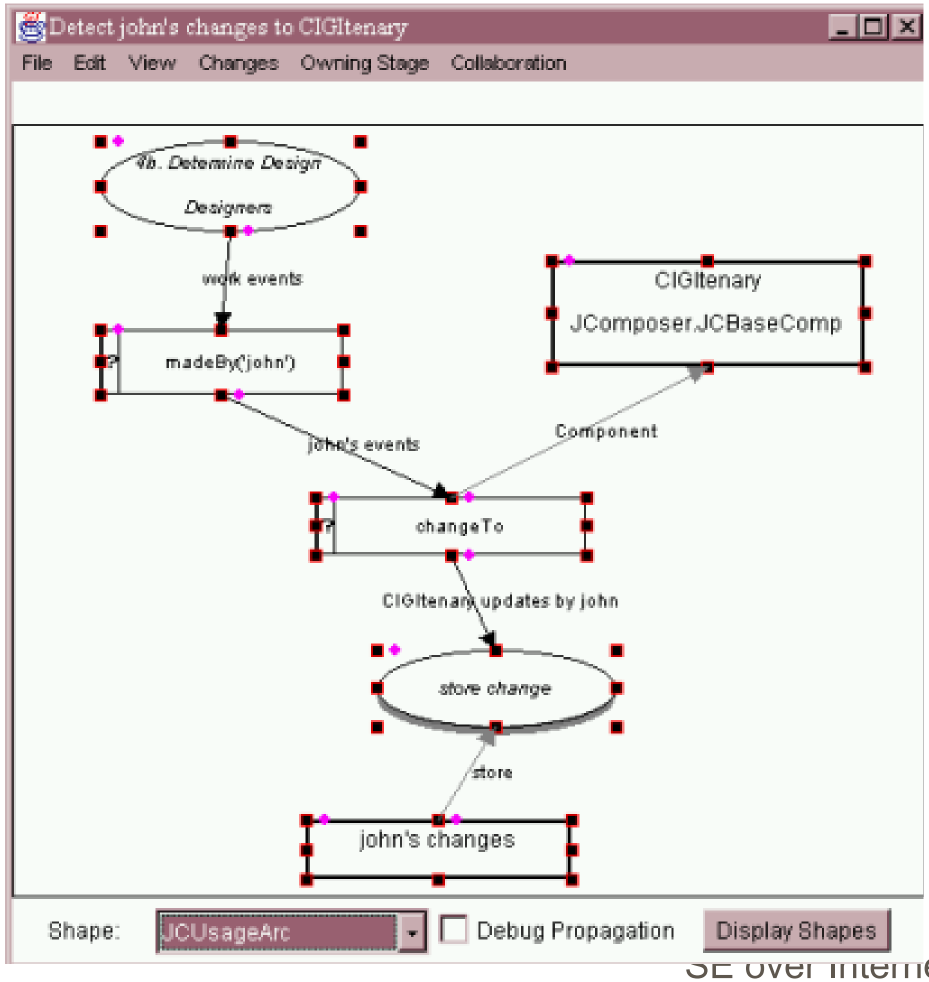


Received changes from john...

[john] 30 Macro change: Moved window
[john] 31 Macro change: Move CIGItenaryItem
[john] 32 Macro change: Move CIGItenaryEditor
[john] 33 Macro change: Add Link
[john] 34 SetStringValue :nameText to store
[john] 35 SetStringValue :nameText store to sto
[john] 36 Macro change: Move MVVersionRecor

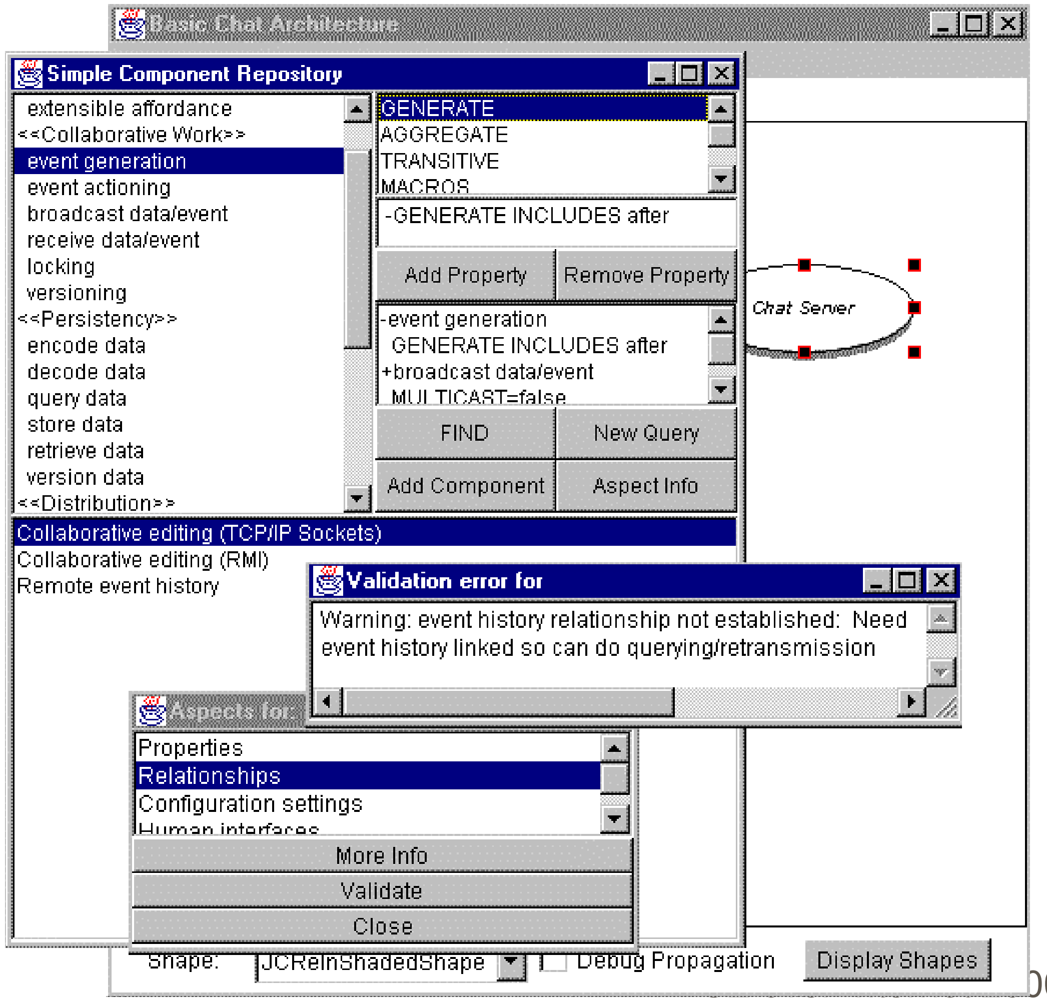
- Plugged into all of our tools
- This example: JComposer component CASE tool/ programming environment
- Sync vs async editing
- Various levels of "sync"
- Versioning
- Group awareness

Distributed Task Automation Agents



- Built using Serendipity-II facilities
- Compine filters/actions to specify co-ordination agent behaviour
- Can deploy locally or on other's machines
- Can subscribe to remote events/ invoke remote component functions

Shared Component Library



extensible affordance
<<Collaborative Work>>
event generation
event actioning
broadcast data/event
receive data/event
locking
versioning
<<Persistency>>
encode data
decode data
query data
store data
retrieve data
version data
<<Distribution>>

GENERATE
AGGREGATE
TRANSITIVE
MACROS

-GENERATE INCLUDES after

-event generation
GENERATE INCLUDES after
+broadcast data/event
MULTICAST=false

Collaborative editing (TCP/IP Sockets)
Collaborative editing (RMI)
Remote event history

Warning: event history relationship not established: Need event history linked so can do querying/retransmission

Properties
Relationships
Configuration settings
Human interfaces

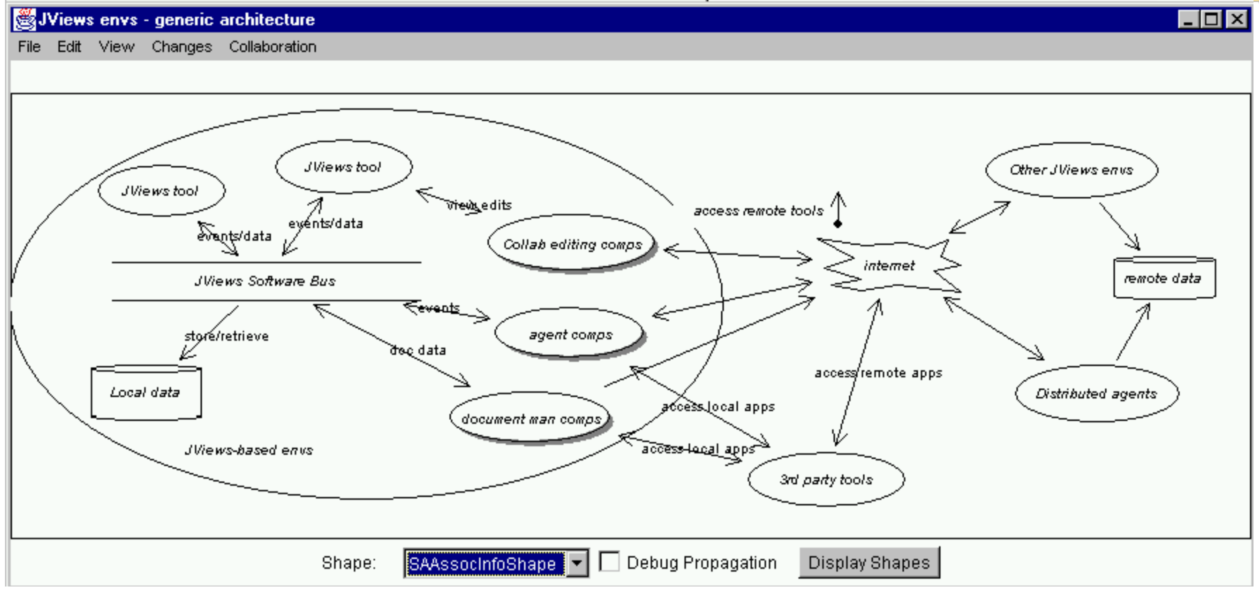
Shape: JCRelnShadedShape

- Components characterised by "aspects"
- Stored & indexed by aspect
- Retrieved by aspect-based queries
- Users can examine/validate component aspects
- Newly created comps added to tool e.g. Ser-II, JComposer, SoftArch...

Assessment

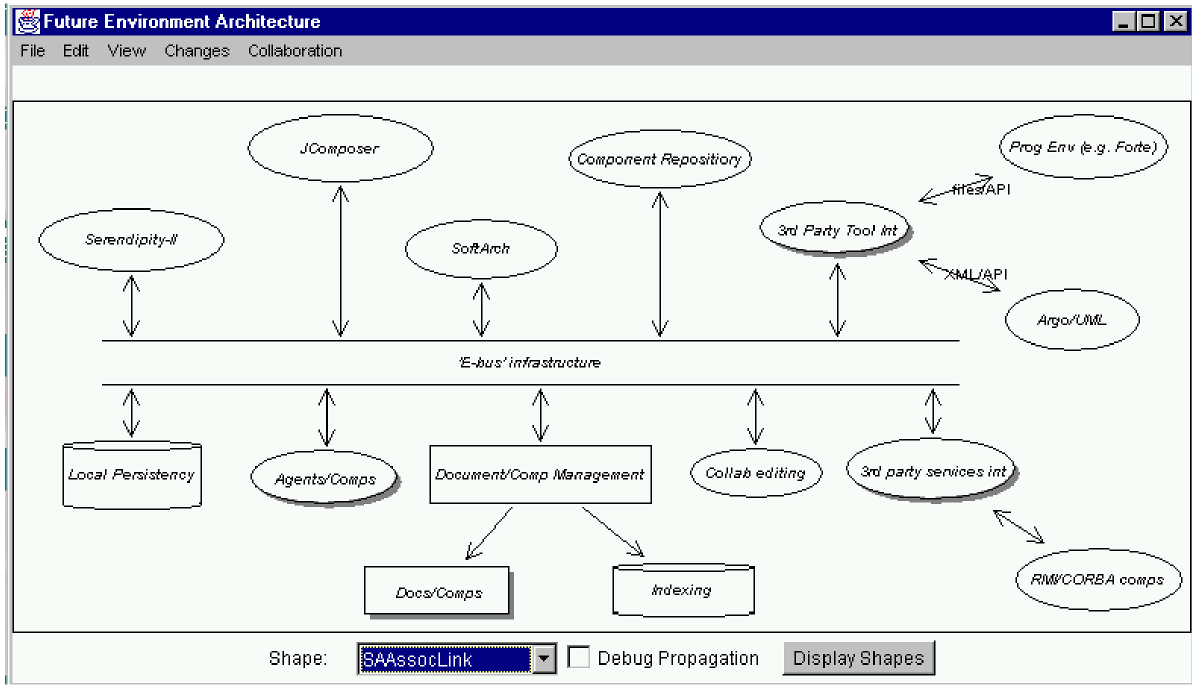
- Provides range of component engineering facilities
- Used to support “Aspect-oriented Component Engineering” method
- Almost all tools/agents reused
- Deficiencies:
  - lack of seamless 3rd party tool support
  - distributed component testing/debugging
  - component/application deployment

Current Architecture...



- Local tools integrated via JViews "event bus"
- Local data stored in files/DB
- Plug-in comps for e.g. agents, document editing, versioning etc.
- Limited 3rd paty tool integration abstractions
- Library/remote agents - limited management/interfaces

E-Bus



- Transparent local/remote event subscribe/notify
- Persistency/distribution transparent
- Agents, document management, collab editing etc all transparent
- Better 3rd party integration
- Same teatment of vertical (tool) and horizontal (services) components…
- Links to bus may be complex comp/data mappings

Summary

- Developed variety of tools (using component-based framework)
- Integrated to support distributed component development
- New facilities:
  - remote component debugging/testing (JVisualise++)
  - further reusable groupware components
  - Better high-level comp/comp group reuse in SoftArch
- New tool development:
  - New fully event-based infrastructure

References

- Grundy, J.C., Mugridge, W.B. and Hosking, J.G. Constructing component-based software engineering environments: issues and experiences, Information and Software Technology Vol 42, No. 2, Special Issue on Constructing Software Engineering Tools, Elsevier Science Publishers.
- Grundy, J.C. A method and environment for distributed component engineering, In Proceedings of the 2000 Conference on Software - Methods & Tools, Wollongong, Australia, Nov 6-10, 2000, IEEEE CS Press.
- Grundy, J.C. Distributed Component Engineering using a Decentralised, Internet-based Environment, In Proceedings of the 3rd ICSE Workshop on Software Engineeering over the Internet, ICSE 2000 Workshop, Limerick, Ireland, June 6 2000, pp. 20-29.
- Grundy, J.C. Visual specification and monitoring of software agents in decentralised process-centred environments, International Journal on Software Engineering and Knowledge Engineering, Vol. 9, No. 4., August 1999, World Scientific Publishing Company, pp. 425-444.
- Grundy, J.C. Aspect-oriented Requirements Engineering for Component-based Software Systems, 1999 IEEE Symposium on Requirements Engineering, Limmerick, Ireland, 7-11 June, 1999, IEEE CS Press.
- Grundy, J.C. Engineering component-based, user-configurable collaborative editing systems, Engineering for Human-Computer Interaction, Chatty, S. and Dewan, P. Eds, February 1999, Kluwer Academic Publishers.
- Grundy, J.C., Hosking, J.G., Mugridge, W.B., Apperley, M.D. A decentralised architecture for software process modelling and enactment, IEEE Internet Computing: Special Issue on Software Engineering via the Internet, Vol. 2, No. 5, September/October 1998, IEEE CS Press, pp. 53-62.
- Grundy, J.C., Hosking, J.G., Mugridge, W.B. Inconsistency Management for Multi-view Software Development Environments, IEEE Transactions on Software Engineering: Special Issue on Managing Inconsistency in Software Development, Vol. 24, No. 11, 1998, IEEE CS Press.