OK so firstly let’s change the emphasis of my presentation slightly.

Doug Schmidt yesterday presented pretty comprehensively technology trends in our space.

But how do these technology trends impact a Company like PrismTech (a middleware and tools vendor) and how do we adapt our business model to be successful for example:

- Open Source – Do we embrace it or fight it?
- Standards – Be committed or complain from the sidelines?
- Funding Model – Profitable Organic Growth or Venture Funding?
- Strategic Choices – Go Vertical or Horizontal?
- Etc. etc.

So a snapshot of PrismTech……….
PrismTech Snapshot

- Established, growing, profitable business
- Telecom, Defense & Financial “Fortune 500” client base in USA and Europe and Asia Pac
- Fast growing productivity tools and middleware base including CORBA, Security products and now Open Splice DDS middleware, big ongoing MDD investment
- 100 people operating in the Europe and the US

High growth strategy aiming for leadership position in the emerging market for wireless infrastructure software and tools for Software Defined Radio (SDR) applications, a segment of the overall SDR market
Our Business……

How do we run our business?
- Ethical
- High quality

What do we deliver?
- Standards based Software tools, middleware & supporting services.

Who are our customers?
- IT professionals
- Large telecom, financial & defense/aerospace industries

Why do they buy?
- An exceptional high quality team
- Exceptional ROI thru rapid development & deployment
- High quality, robust, innovative applications and products
- Lowest total cost of ownership
Our Technology Investments

- Software Defined Radio MDD Productivity Tools and Middleware
- DDS with OpenSplice, DCPS, DLRL and MDD Tools
- By adding value to middleware solutions in specific application domains (e.g. Telco, Mission Systems) e.g. Security functionality to domain solutions
- In CORBA/e and in extending our MDD tooling into the embedded domain
- RT Java initially running with Sun and IBM
- MDD with wider Domain Specific Tooling
- In extending our commercial support offerings around Open Source Middleware (e.g. Ciao)
A unique synergistic blend of horizontal and vertical domain expertise

**Horizontal**
- Hardware
- OS
- DRE Middleware
- Components
- Model Driven Engineering

**Vertical**
- Software Define Radios
- Missions Solutions
- Security
PrismTech Experts at the OMG RTE Conference

- Sebastian Staamann – Security
- Bruce Trask - MDD
- Dominick Paniscotti - SDR
- Hans van’t Hag – DDS
- Simon McQueen - CORBA
2005 - Bruce And Dom (and Jerry) came blazing into PrismTech

But The Real Question… Why Does Bruce Get to Be Robert Redford?
From May 1st A New PrismTech DDS Team in Hengelo

Hans van 't Hag

Hans was Product Manager, SPLICE-DDS at Thales Naval Netherland and as middleware architect was responsible for the development of the data-centric real-time distributed middleware (SPLICE) as applied in Thales Naval Netherland's naval Combat Management Systems. He has over 10-year experience of applying an information-centric approach towards mission-critical and real-time systems into emerging Open Architecture standards such as the OMG-DDS Data Distribution Service for real-time systems.

He has 23 year experience as engineer, architect, group-manager, 'MWR-technical authority', and product manager

Other DDS Hengelo team members

Robert Poth
- DDS architect, 16 years of middleware/DDS related experience, 6 years as overall SPLICE architect

Reinier Torenbeeks
- Expert-level engineer with >9 year experience & SPLICE-DDS network architect

Hans Venema
- Expert-level engineers with >7 year DDS experience

Erik Hendriks
- Expert-level engineers with >7 year DDS experience

Niels Kortstee
- Engineer, 3 years of working DDS experience
2006 - The SPLICE Boys ride into town...

Scary  Posh  Baby  Ginger  Sporty

More Later……..
PrismTech is aggressively developing a standards compliant, high performance and reliable COTS implementation of the OMG’s Data Distribution Standard (DDS).

Recently hired team of leading DDS domain experts in order to complement existing PrismTech engineering team lead by Hans Van’t Hag (former Thales Netherlands).

Have now agreed deal with Thales for PT to further develop and commercially exploit SPLICE as full COTS offering to be known as OpenSplice, which consists of the following major components:

- OpenSplice Developer Productivity Tools
- OpenSplice DLRL
- OpenSplice DCPS (a.k.a. SPLICE-DDS)
Embrace Open Source

- We have embraced Open Source since 2001 with a strategy based on adding industrial strength to TAO and JacORB

- By applying COTS software product principles to leading Open Source offerings

- A model that customers understand and prefer:
  - Packaged solution
  - Formal testing and QA processes
  - Worldwide support & maintenance (including 24x7)
  - Training & professional Services
  - Product roadmap including
    - Customer enhancements
    - Operating System updates

- …..but with NO licence fees
PrismTech’s Added Value to Open Source

- **Packaged distributions**
  - Additional functionality and fixes not available from Open Source community
  - Pre-built binary distributions supporting a large number of Operating Systems
  - Additional examples (fully documented)
  - Easy installation and configuration
  - Additional documentation
    - Comprehensive release notes
    - User documentation

- **Version & release management**
  - Major releases synchronised with community releases
  - Patch releases in response to customer reported issues or in response to fixes created by the community
  - Frozen and extended version support
  - Formal testing and QC for each release
    - Including dedicated testing to help ensure interoperability between TAO, JacORB and OpenFusion CORBA Services
  - Co-ordination with community over development and release activities

- **Commercial-Grade Support**
CORBA Middleware for RTE - Misconceptions

- CORBA based Middleware
  - Too big …
  - Too slow …
  - Too heavyweight …
  - Not designed for small scale systems …

- PrismTech Response:
  - It’s possible to run the next generation of CORBA middleware efficiently in even the most resource constrained environment such as a DSP or FPGA and we can prove it…….but OMG…

- The CORBA name is proving more of a hindrance than a help
  CORBA/e is a valid way forward – ‘CORBA for me’ but the time may now be right for a more radical move – a new brand for embedded standard distributed middleware
Stop Having the Tail Wag the Dog

GPP - Enterprise
- Large memory capacity
- Rich system resource support
- Operating Systems supports POSIX APIs

GPP - Embedded
- Limited memory capacity
- Basic system resources available (tasks, semaphores, mutex, condition variables)
- Operating Systems may support POSIX APIs

DSP
- Very constrained memory capacity < 2Mb
- Basic system resources available (tasks, semaphores, locks, interrupts)
- Usually no POSIX APIs

FPGA
- Small amount of logic in FPGA
- Implemented CORBA features to support SCA
- Low level transport layer
Next Generation CORBA on DSP & FPGA

- **DSP platform**
  - Significantly reduced footprint ORBs can fit in on-chip memory.
  - C ORB deployment < 100k (TI TMS320V5510)
  - Modular ORB functionality
    - Pick ‘n’ Mix approach to building ORB function
  - Pluggable transport layers allowing smaller and faster transport implementations
    - ETF based transport TCP, UDP, Shared Memory, Rapid I/O, Raw Ethernet
  - Pooling of system resources (threads, memory)
  - System characteristics obeying real-time requirements

- **FPGA platform**
  - ICO an Integrated Circuit ORB

- These products are all here now!
Users blame poor performance on middleware

However, transport is typically the limiting factor

Transport choices are very limited

Custom transports typically home grown by experts

Not very many COTS choices available

ETF can correct this problem, making transports easier to build and easier to move from ORB to ORB

Incorporating new transports is inherently complex
Platform technologies outpacing languages

CORBA language bindings difficult to use?

Are they? Yes!

Does it matter? No!

Are new CORBA language bindings warranted? No!

Assembly Language is difficult to program in also

Was it eliminated? No!

Replacement of language bindings is not the answer to solving CORBA complexity

Raising language abstraction levels is!
No Silver Bullet (Fred Brookes)

Systems contain *inherent* and *accidental* complexities

Inherent Complexities
Those found within the problem domain itself

Accidental Complexities
Those accompanying the production of software but can be eliminated

Elimination of the accidental complexities allows developers to focus on the inherent complexities
CORBA users have lots of inherent complexities!

- Object Oriented
- Component Based
- Multithreaded/MultiProcess
- Real-time
- Embedded
- C++/Java/Ada/VHDL
- Platform Independent
- High Performance
- Heterogeneous
- Distributed
- Vital
- Secure
- Fault Tolerant
- Portable
- Standardized
- Declarative
- Imperative

So what’s the big deal? Each one by itself is difficult, let alone doing them all at the same time
So let’s eliminate Accidental Complexities!

- CORBA Accidental Complexities
  - Performance Tuning
  - Language Bindings
  - Pluggable Protocols

- Early attempts to raise the abstraction levels were slow to be accepted (CCM, D&C, SWRADIO)
  - Much of the focus, again on raising the platform abstractions
  - The language… still difficult to use
    - More accidental complexities (e.g. CIDL, XML Descriptors, etc.)
  - Users must still manipulate middleware settings via 3GLs

- Enter Model Driven Engineering…
DSLs allow simplified modeling in the **Problem Space** vs. complex modeling **in the Solution Space**
“Declarative programming improves productivity and quality because it is another form of reuse of preprogrammed, prevalidated logic”¹

Model Driven approached allow the middleware language binding complexities to be hidden in the transformation rules/generators

Thereby eliminating accidental complexities and increasing architectural consistency, quality, and productivity

¹ Model Driven Architecture D. Frankel, Wiley 2003
MDE/MDA holds the promise of solving all of the issues described above

Model Driven Domain Specific Tools
- Raise the abstraction levels
- Unshackle developers from the accidental complexities of middleware
- Allows them to focus on inherent complexities of their systems
- Puts new graphical languages on par with platform abstractions
- Allow code generation of 3GL artifacts and descriptors automatically
- Allow users to choose 3GL to generate to support their computing environments (resource constrained, real-time, fault tolerant, etc.)
With the release of its Software Defined Radio modeling tools (Spectra), PrismTech sets the stage for model-driven middleware integrated development.

PrismTech is leveraging this work to support model-driven middleware tools for DDS (OpenSplice).

Using these tools and hence eliminating the accidental complexities of manipulating middleware technologies in “by hand” increases development times many fold, via automatic:
- Middleware source code and descriptor generation
  - In language choice
- Middleware policy generation
- Performance tuning parameter generation
- Build file generation
- Etc, etc, etc.
<table>
<thead>
<tr>
<th>Catch Phrase</th>
<th>System</th>
<th>Collections</th>
<th>Components</th>
<th>Scale</th>
<th>When/Peak</th>
<th>Leaf Protocol(s)</th>
<th>Directory(s)</th>
<th>Session</th>
<th>Schematic</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Network Is the computer</td>
<td>Objects</td>
<td>Legacy to the Web</td>
<td>The Computer Is the Network</td>
<td>Network of embedded things</td>
<td>Network of Things</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100s</td>
<td>1000s</td>
<td>1000000s</td>
<td>10000000s</td>
<td>100000000s</td>
<td>1000000000s</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>+HTTP (+JVM)</td>
<td>+UDDI</td>
<td>+jini</td>
<td>Unknown</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NIS, NIS+</td>
<td>+CDS</td>
<td>+LDAP (*)</td>
<td>+SOAP, XML</td>
<td>+RMI/jini</td>
<td>+?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPC, XDR</td>
<td>+CORBA</td>
<td>+CORBA, RMI</td>
<td>+RMI/jini</td>
<td>+?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"Information - Backbone" (DDS)
Distributed Systems Trends

- **Everything becomes ‘connected’**
  - Dynamic, loosely coupled
  - Traditional (client/server) architecture don’t suffice: too static

- **Everything is becoming part of a ‘system-of-systems’**
  - Ubiquitous computing
  - Spontaneous networking

- **Pub/Sub?**
  - Asynchronous nature → suitable pattern
  - Yet until shortly, no standard available

- **DDS**
  - Combines P/S with extensive QoS
  - Applicable for mission-critical & constrained environments

- **OpenSplice**
  - PT’s full implementation of this new standard
  - Uniquely covering both information-distribution & Information management
DDS facilitates a complexity shift
- From application to middleware level
- Information bus: ‘right information, right place, right time’

DDS Domain specific tooling required to master DDS complexity
- System-level
  - ‘model’ the spoken language of components (semantics, syntax, QoS)
- Application level
  - ‘model’ the application-structure (allow programmers to concentrate on BL)
- Deployment level
  - ‘model’ the information-bus (networking/processing resources)
  - Control & monitor deployed systems (tuning, maintenance)

So: an Innovative approach
- New category of networked-systems requires new tooling
- Separation-of-concerns to master complexity
- Challenge: ‘roundtrip-engineering’ from one integrated MDD-suite
PT ready to ‘take-up’ the challenge

Working towards ‘first-in-class’ DDS MDD tool-suite
- As necessary tool-suite to ‘unleash’ the full potential of OMG-DDS
- As vital component in a total round-trip engineering solution

Facilitating ‘information-centric’ distributed systems engineering
- Clearly distinguishing between lifecycle phases and abstraction-levels
  - Information modeling, application modeling, deployment modeling
- Supporting autonomous components
  - Communication/interaction ‘taken care of’ by DDS ‘Information Backbone’
- Providing round-trip engineering
  - ‘manage’ complexity
  - ‘manage’ evolution

More to follow this afternoon….
OpenSplice is an implementation of the OMG Data Distribution Service specification, with supporting tools facilitating its usability in a variety of system environments.

- **Core module: Pub-Sub Engine**
  - Minimum + Ownership Profiles
- **Information Management modules:**
  - content subscription
  - Persistence
- **Object model module (DLRL)**
  - OO-view on relational topic data
  - Relationship management and object-navigation
- **Productivity toolSuite**
  - OpenSplice Information Modeller
  - OpenSplice Application Modeller
  - OpenSplice System Tuner
- **Value-add extensions and tools**
  - Language bindings (e.g. Ada)
  - Corba cohabitation
  - DDS interoperability protocol
Security for DRE...

Date: 11th May 2006
New challenges in the area of security have appeared

- Distributed Real-time Embedded (DRE) systems often used in very security critical application areas (e.g., Defense):
  *Nowadays DRE systems must be complemented with appropriate security products and solutions*

- Open architectures (OA) for the internal structure and the cooperation with other DRE systems (Systems-of-Systems):
  *The OA computing environment must contain security components that enable the systems designers and the operators full security control of the interactions between systems (potentially from different suppliers)*

- Use of various standardized distribution middleware (CORBA, DDS, XML Web Services, Group Communications) in each system:
  *The security technology must enable the definition, enforcement, and management of system-wide unified security policies at the application layer regardless which middleware technology is used.*

- Model driven design of DRE systems:
  *The model driven design of the systems and applications must include the modeling of the formal security policies to be enforced in the operation phase.*
Competition is not only good it’s essential

- Competition makes a market
- A single implementation of a standard is the same as a proprietary solution, Corporations will NOT embrace your technology (unless you’re Microsoft)
- There is NOT enough Open competition in the Embedded Middleware space
- DDS shows the way
- All suppliers co-operating to help provide INDEPENDENT comparative performance data, this shortens sales cycle time and gives the market confidence in the technology
- We need the same approach in embedded CORBA
Contact Information

PrismTech
www.prismtech.com

Keith Steele
keith.steele@prismtech.com

Thank You ....