Remedy IT

Your challenge - our solution

Modernizing SCA through new Object Management Group (OMG) standards

Johnny Willemsen (jwillemsen@remedy.nl)

CTO Remedy IT

http://www.remedy.nl

Remedy IT

Your challenge - our solution

OMG Standards

- SCA can be modernized through several OMG standards initiatives
  - Available standards
    - IDL to C++11
    - IDL4
  - Upcoming standards
    - Unified Component Model for Distributed, Real-Time, and Embedded Systems (UCM)

Remedy IT

Your challenge - our solution

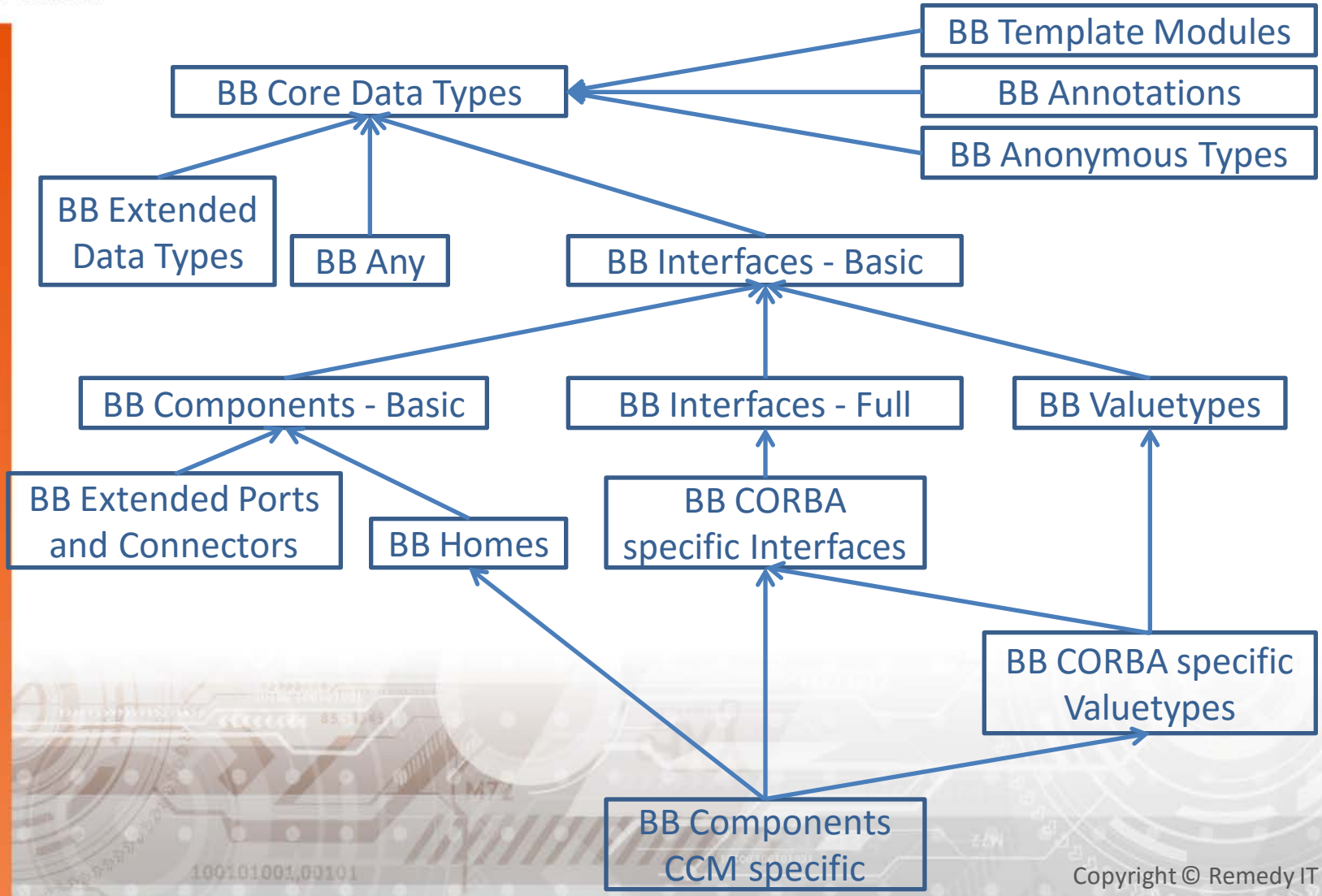
IDL4

- The IDL4 specification performs the following actions
  - Moves IDL out of the CORBA specification and into its own OMG specification
  - Integrate the IDL extensions coming from the DDS X-Types specification
    - Annotations, map, bitset
  - Provides a logical grouping of IDL constructs into different building blocks (BB)

IDL4 building blocks

Remedy IT

Your challenge - our solution



Remedy IT

Your challenge - our solution

IDL4 and SCA

- IDL4 simplifies SCA with minimal efforts
  - Refer to the new IDL4 building blocks instead of the full CORBA IDL chapter
  - Remove all text about which parts of the CORBA IDL chapter can be ignored

Remedy IT

Your challenge - our solution

Unified Component Model for Distributed, Real-Time, and Embedded Systems

- New software component standard
- Evolution of LwCCM, RTC, SCA, and related efforts
- Platform independent/specific model approach
- Software communication middleware agnostic
  - No mandatory dependency on CORBA or any other communication middleware standard
- Will provide an IDL4 PSM
  - All interfaces are local by default

Remedy IT

Your challenge - our solution

UCM Interaction Patterns

- Interaction Patterns define how components interact with the outside world
  - Request/Reply interaction
    - client, server, asynchronous client, and asynchronous server
  - Event interaction
    - supplier, push consumer, and pull consumer
  - State interaction
    - observable, passive observer, push observer, pull observer, and push state observer

Remedy IT

Your challenge - our solution

UCM Connector Fragments

- Connector fragments realize a specific interaction pattern role
  - Sockets, DDS, CORBA, http, …
  - Infiniband, serial, …
  - Existing systems, non-CBDDS systems
- Implemented or generated by you as user or by a vendor

Remedy IT

Your challenge - our solution

IDL to C++11

- Simplified mapping for C++
  - Make use of the standard C++ library as much as possible, no CORBA throughout the application code
- Make use of the C++11 features to
  - Reduce amount of application code
  - Reduce amount of possible coding errors by providing a safer API
  - Gain runtime performance
  - Speedup development and testing
    - Faster time to market
    - Reduce costs
    - Reduce training time

Remedy IT

Your challenge - our solution

Basic Types

IDL	C++11	Default value
short	int16_t	0
long	int32_t	0
long long	int64_t	0
unsigned short	uint16_t	0
unsigned long	uint32_t	0
unsigned long long	uint64_t	0
float	float	0.0
double	double	0.0
long double	long double	0.0
char	char	0
wchar	wchar_t	0
boolean	bool	false
octet	uint8_t	0

Remedy IT

Your challenge - our solution

Reference Types

- No ptr/var/duplicate artifacts anymore but so called reference types
  - Automatically reference counted
  - Nil reference is represented as `nullptr`
  - A boolean operator for comparison is available
  - Invoking an operation on a nil reference results in an exception, no need to check every invocation of a reference
- Instead of defining all kinds of naming rules all implied C++ reference types are available through `IDL::traits<>` including additional meta information as defined in your IDL!

Remedy IT

Your challenge - our solution

Other Types

- A lot of improvements and simplifications, including
  - IDL string is mapped to `std::string`
  - IDL sequences are mapped to `std::vector`
  - IDL arrays are mapped to `std::array`
  - IDL exceptions are derived from `std::exception`
  - IDL enums are mapped to C++11 strongly typed enums
  - IDL structs and unions are both mapped to C++ class
  - Simplified argument passing

Remedy IT

Your challenge - our solution

IDL to C++11 and SCA

- IDL to C++11 isalready referred by the SCA IDL PSM
- Application code using IDL to C++11 and the SCA IDL PSM doesn’t use CORBA specific types
- Easy to use will lower the costs and time to market of SCA based systems
- Next step is to find/define a proof -of-concept project to demonstrate IDL to C++11 to the SCA community, looking for interested parties!

Remedy IT

Your challenge - our solution

Want to know more?

- Check our website at http://www.remedy.nl
  - TAOX11, CORBA implementation supporting IDL to C++11
  - AXCIOMA, the component framework for distributed, real-time, and embedded systems
  - RIDL, open source IDL compiler front end

Remedy IT

Your challenge - our solution

Contact

Remedy IT
Melkrijder 11
3861 SG Nijkerk (Gld)
The Netherlands

tel.: +31 88 053 0000
e-mail: sales@remedy.nl
website: www.remedy.nl

Twitter: @RemedyIT
Slideshare: RemedyIT

Subscribe to our mailing list