DEFENCE R&D DÉFENSE

# SOCLe:
# *Integrated Design of Software Applications and Security*

Frédéric Painchaud

Defence Scientist

Trusted C2IS Group

R et D pour la défense Canada
Defence R&D Canada

Canada

# High-Assurance Software: A Question of Correct Design

- Software abounds in mission-critical systems
  - Banks, hospitals, emergency services, public utilities, governments, military organizations, etc.
- High reliability and security requirements
  - But very complex and fragile
- Three main issues:
  - Design problems (system architecture)
  - Implementation problems (programming languages and platforms used)
  - Malicious code (viruses, trojans, worms, backdoors, etc.)
- Correcting errors during or after implementation is costly
  - 50 to 100 times more expensive than at design-time

Source: B. W. Boehm and P. Papaccio. Understanding and controlling software costs. In *IEEE Transactions on Software Engineering*, volume 14, pages 1462-1477, October 1988.

# Validating Design: Our Key Approach

- Integrate design and system specification
- Use an innovative mix of UML and OCL

# UML and OCL

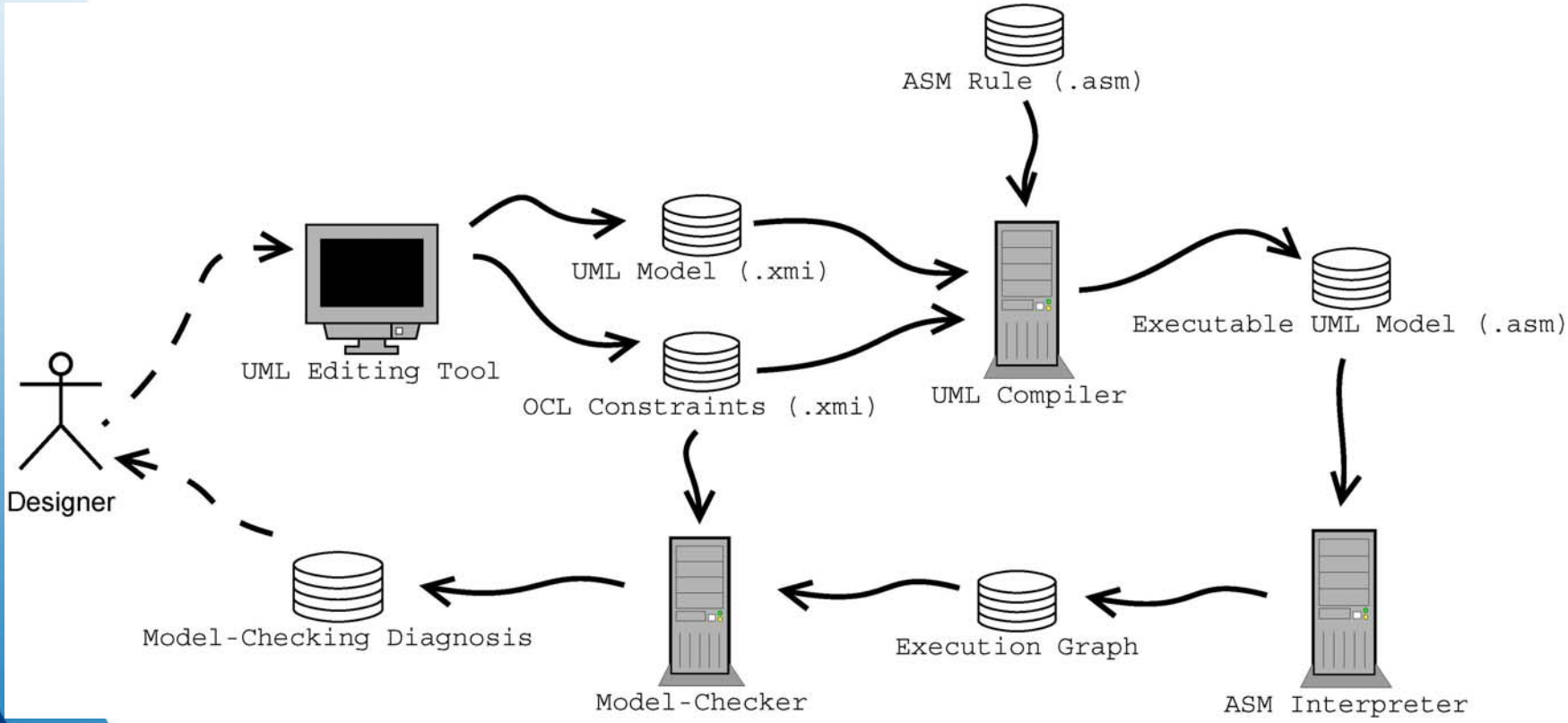
- UML is the *de facto* standard software design notation
  - Used in the industry and in C2IS and other governmental critical systems
- OCL is a complementary constraint language
  - Eliminates ambiguities in software design
  - Formulates pre-/post-conditions and invariants
  - Part of UML

UML: Unified Modeling Language
OCL: Object Constraint Language

# Plan of the Presentation

1. **SOCLe Architecture**
2. Caveat-Separation System: A Simplified Case Study
3. SOCLe Prototype
4. Conclusion

# SOCLe Architecture



# Plan of the Presentation

1. SOCLe Architecture
2. **Caveat-Separation System: A Simplified Case Study**
3. SOCLe Prototype
4. Conclusion

# Caveat-Separation System: A Simplified Case Study

- Users
  - **Security clearance**: Enhanced, Confidential, Secret, or Top Secret
  - **Country of origin**: Canada, United States, United Kingdom, Germany, Egypt, etc.
  - **Military rank**: General, Officer, or Civilian
- Documents
  - **Classification level**: Unclassified, Confidential, Secret, or Top Secret
  - **Caveat**: CEO, CANUS, CANUSUK, NATO, UN Coalition, USUK, etc.
  - **Affiliation with a project**: A, B, C, etc.

# Caveat-Separation System: A Simplified Case Study

- Access Control Tables
  - **Classification table**: access relation between the security clearance of users and the classification level of documents
  - **Caveat table**: caveats dictate acceptable countries of origin for users who want to access documents
  - **Need-to-Know table**: determines the need-to-know, in terms of countries of origin and military ranks, that is necessary to access the documents of a particular project
- System property
  - At all times, a user can access a document if and only if the access is granted with respect to the three access control tables.

# Classification Table

| Classification level / Security clearance | Enhanced | Confidential | Secret | Top Secret |
|---|---|---|---|---|
| Unclassified | Granted | Granted | Granted | Granted |
| Confidential | Denied | Granted | Granted | Granted |
| Secret | Denied | Denied | Granted | Granted |
| Top Secret | Denied | Denied | Denied | Granted |

# Caveat Table

| Caveat / Country of origin | Canada | United States | United Kingdom | … |
|---|---|---|---|---|
| CEO | Granted | Denied | Denied | … |
| CANUS | Granted | Granted | Denied | … |
| CANUSUK | Granted | Granted | Granted | … |
| … | … | … | … | … |

# Need-to-Know Table

| Project / Country of origin | Canada | United States | United Kingdom | … |
|---|---|---|---|---|
| A | G – O – C | G – O – C | | … |
| B | G | G – O | G – O – C | … |
| C | G – O | G | | … |
| … | … | … | … | … |

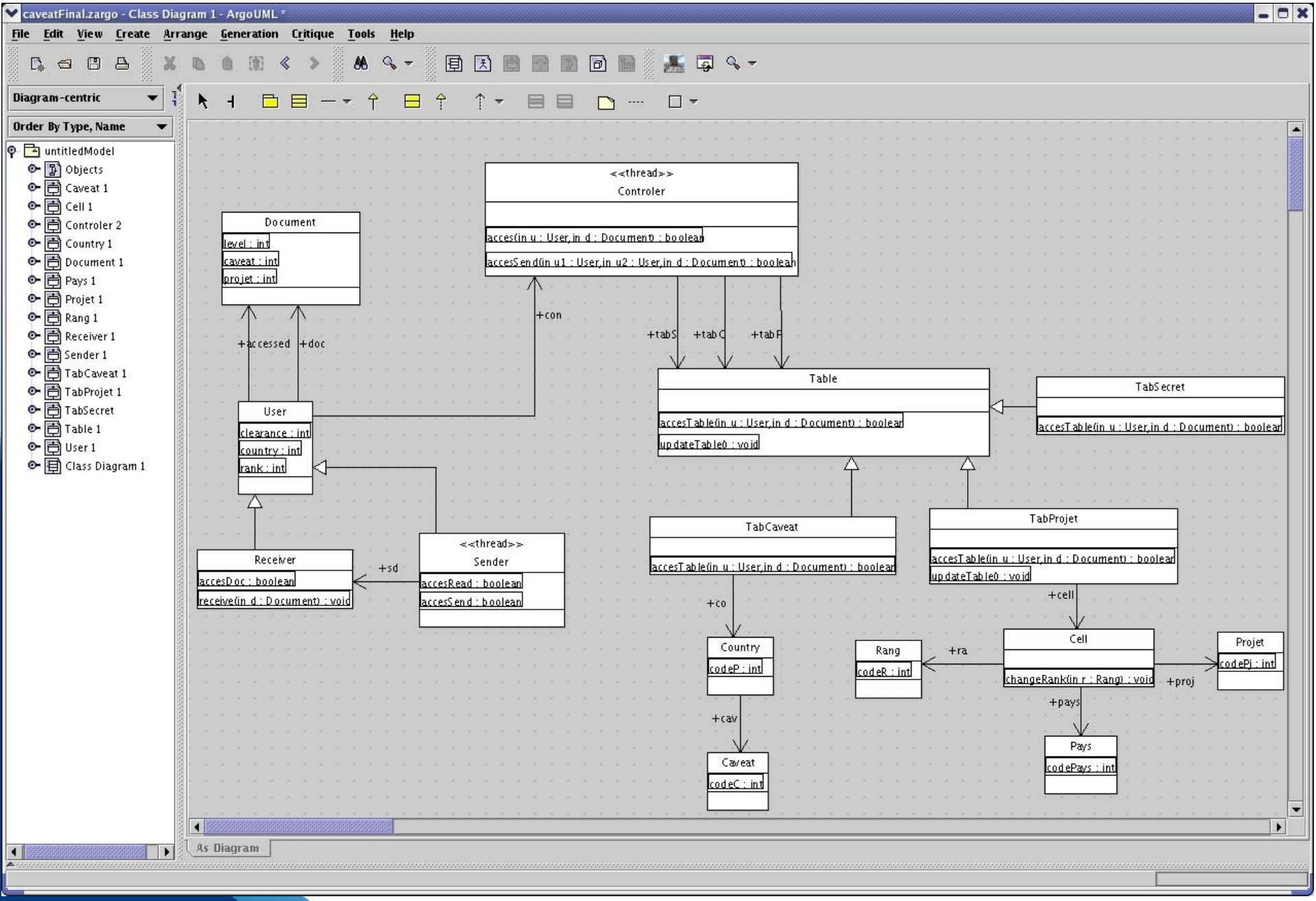
G: General
O: Officer
C: Civilian

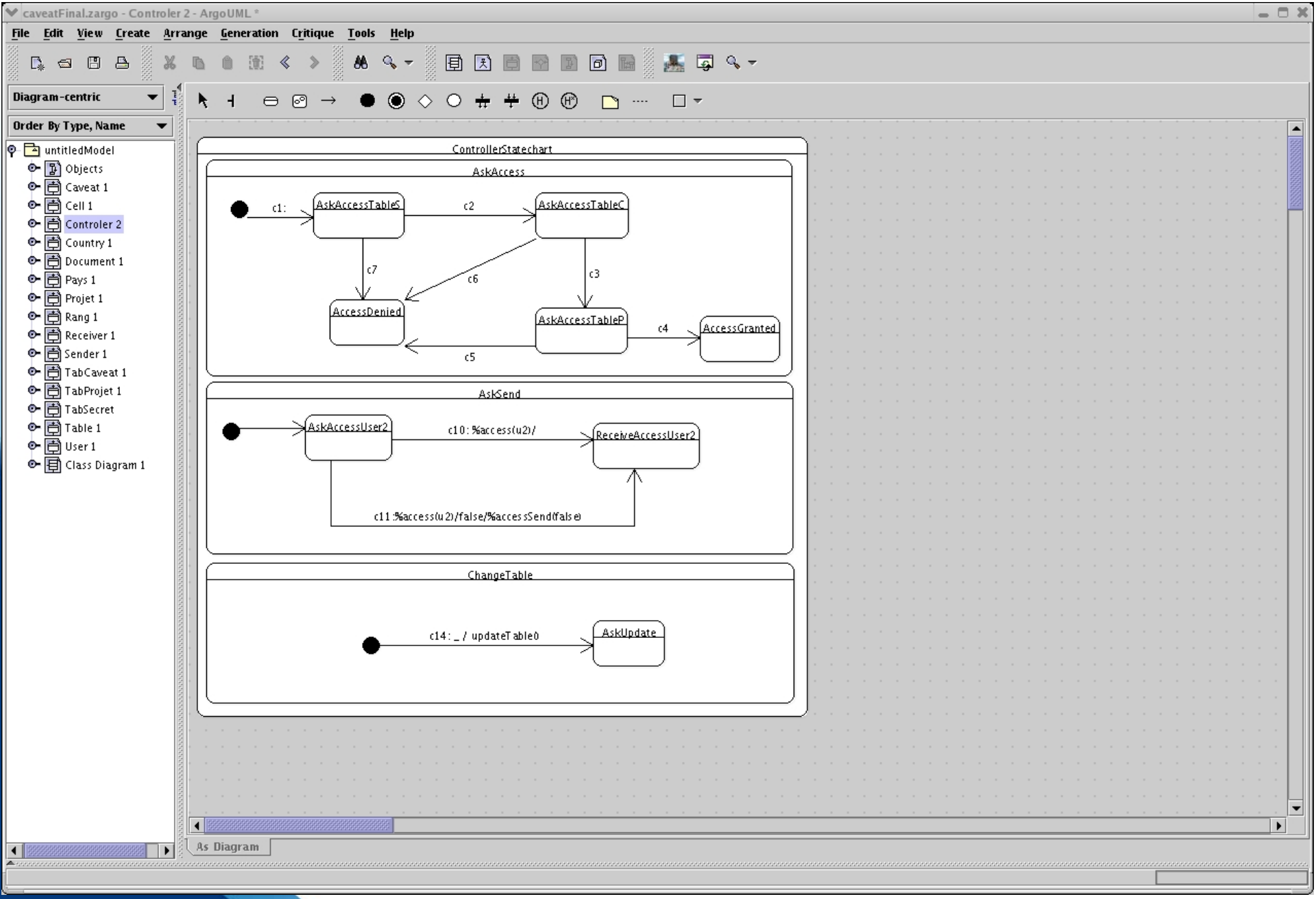
# Plan of the Presentation

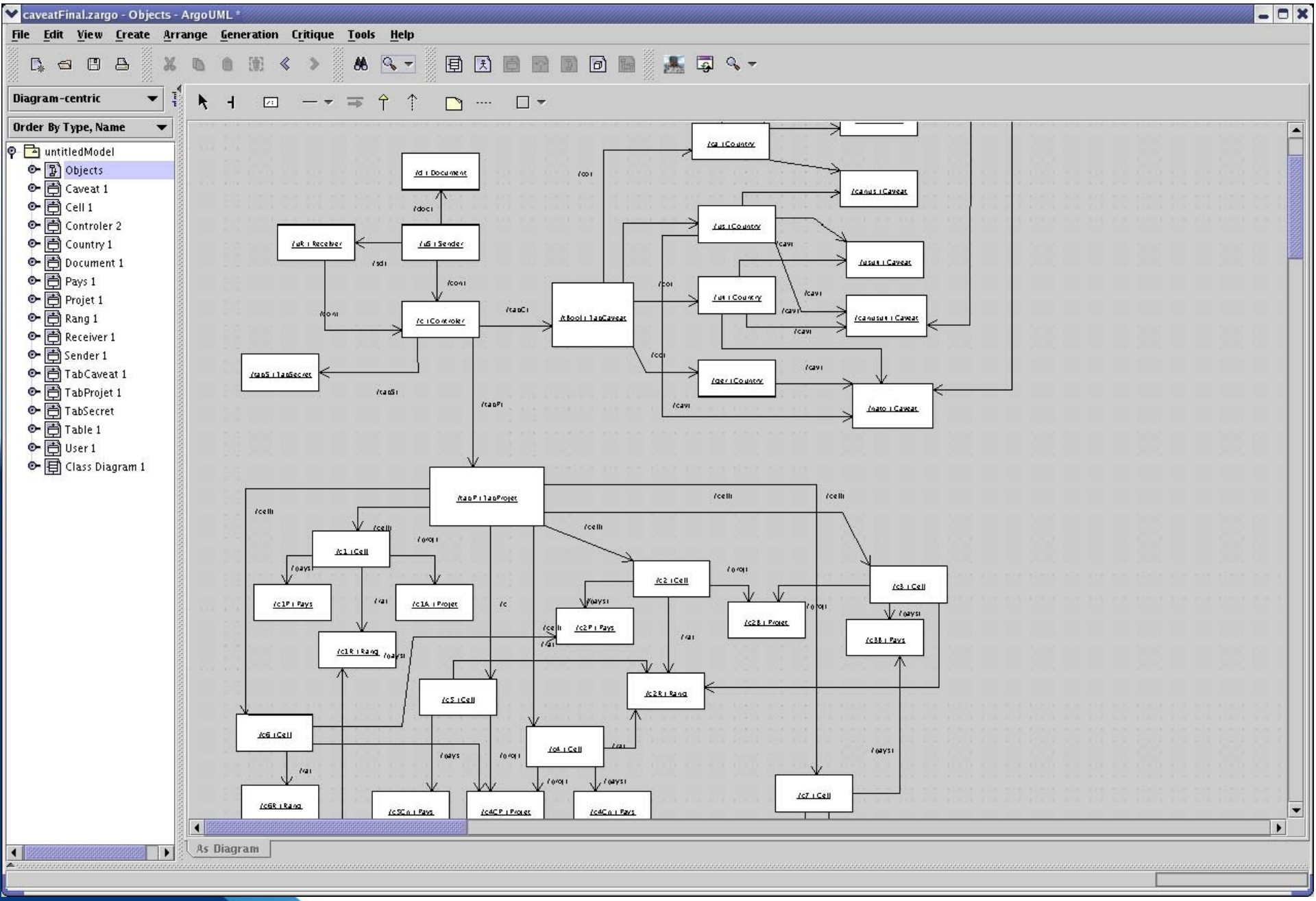
1. SOCLe Architecture
2. Caveat-Separation System: A Simplified Case Study
3. **SOCLe Prototype**
4. Conclusion

# SOCLe Prototype

- Design the system in the usual way
  - Class diagrams, object diagrams, collaboration diagrams, statecharts, etc.
  - Add OCL constraints
- SOCLe constructs an underlying model
  - Formally verified by under-the-hood model-checking
  - Transparent to the designer


caveatFinal.zargo - Class Diagram 1 - ArgoUML *
File Edit View Create Arrange Generation Critique Tools Help
Diagram-centric
Order By Type, Name
untitledModel
Objects
Caveat 1
Cell 1
Controler 2
Country 1
Document 1
Pays 1
Projet 1
Rang 1
Receiver 1
Sender 1
TabCaveat 1
TabProjet 1
TabSecret
Table 1
User 1
Class Diagram 1
<<thread>>
Controler
accesSend(in u1 : User,in u2 : User,in d : Document) : boolean
Document
level : int
caveat : int
projet : int
+con
+accessed
+doc
+tabS
+tabC
+tabP
Table
updateTable() : void
TabSecret
User
clearance : int
country : int
rank : int
TabCaveat
TabProjet
updateTable() : void
+cell
Receiver
accesDoc : boolean
receive(in d : Document) : void
+sd
<<thread>>
Sender
accesRead : boolean
accesSend : boolean
+co
Country
codeP : int
Rang
codeR : int
+ra
Cell
changeRank(in r : Rang) : void
+proj
Projet
codePj : int
+pays
+cav
Caveat
codeC : int
Pays
codePays : int
As Diagram


caveatFinal.zargo - Controler 2 - ArgoUML *
File Edit View Create Arrange Generation Critique Tools Help
Diagram-centric
Order By Type, Name
untitledModel
Objects
Caveat 1
Cell 1
Controler 2
Country 1
Document 1
Pays 1
Projet 1
Rang 1
Receiver 1
Sender 1
TabCaveat 1
TabProjet 1
TabSecret
Table 1
User 1
Class Diagram 1
ControllerStatechart
AskAccess
c1:
AskAccessTableS
c2
AskAccessTableC
c7
c6
c3
AccessDenied
AskAccessTableP
c4
AccessGranted
c5
AskSend
AskAccessUser2
c10: %access(u2)/
ReceiveAccessUser2
c11:%access(u2)/false/%accessSend(false)
ChangeTable
c14: _ / updateTable()
AskUpdate
As Diagram


caveatFinal.zargo - Objects - ArgoUML *
File
Edit
View
Create
Arrange
Generation
Critique
Tools
Help
Diagram-centric
Order By Type, Name
untitledModel
Objects
Caveat 1
Cell 1
Controler 2
Country 1
Document 1
Pays 1
Projet 1
Rang 1
Receiver 1
Sender 1
TabCaveat 1
TabProjet 1
TabSecret
Table 1
User 1
Class Diagram 1
As Diagram

$$
\begin{array}{rcl}
\textbf{context} & : & Controller :: access(u : User, d : Document) \\
\textbf{pre} & : & u.clearance \geq d.level \\
& \wedge & (\textbf{self}.tabC).co \rightarrow \textbf{filter}\{x|x.codeP = u.country\}.cav \rightarrow \\
& & \textbf{exists}\{y|y.codeC = d.caveat\} \\
& \wedge & (\textbf{self}.tabP).cell \rightarrow \textbf{filter}\{x|x.country.codeCt = u.country\} \rightarrow \\
& & \textbf{filter}\{y|y.proj.codePj = d.project\}.ra.codeR \leq u.rank \\
\textbf{post} & : & \textbf{true}
\end{array}
$$

SOCLE Compiling Execution Diagram

Socle Info

```
== Layer 6: Done ==

== Layer 7: Method Constraints ==
        Extracting Constraints (.xmi):
                Class Diagram...                Done
        Elaborating Constraints:
== Layer 7: Done ==


== Generating .asm  File ==


*** Computing .asm file ***
+-----------------------------------------------------+
|     CRAC's ASM-Compiler & OCL-Model-checker v3      |
|    Damien Azambre & Mathieu Bergeron - 2003-2004    |
+-----------------------------------------------------+

[1] Opening caveatFinal.asm... done

[2] Creating caveatFinal directory... done

[3] Parsing input file...
        Parsing OCL Expressions... done
        Type-checking OCL Expressions... done
done in 0.05 s

[4] Computing execution graph...
##################################################################
##################################################################
##################################################################
##################################################################
##################################################################
##################################################################
```

Show previous results | 10,000 | Stop | Cancel

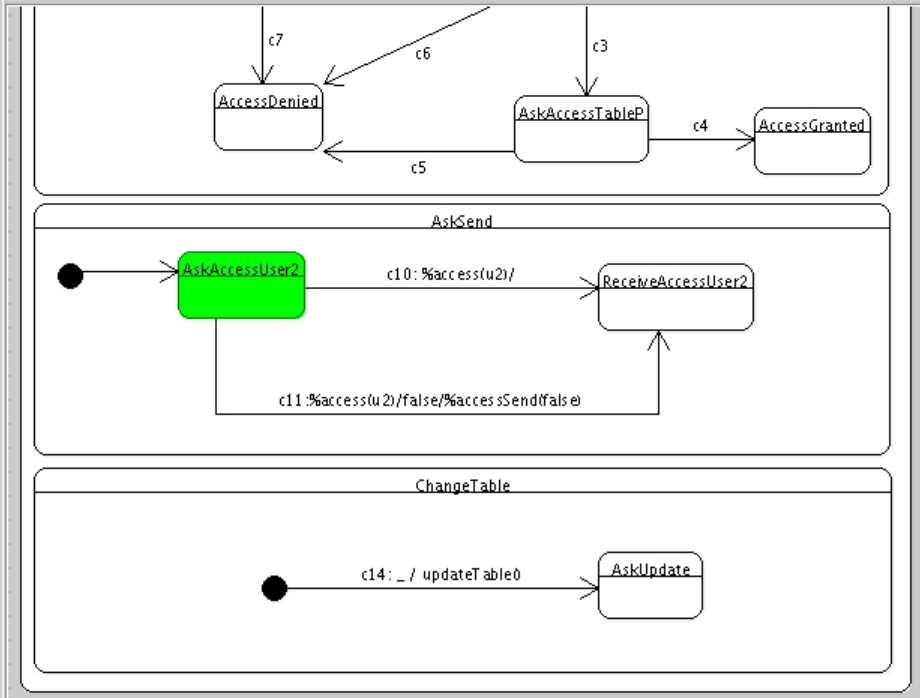
caveatFinal.zargo - Controler 2 - ArgoUML *

File Edit View Create Arrange Generation Critique Tools Help

Diagram-centric

Order By Type, Name

- untitledModel
  - Objects
  - Caveat 1
  - Cell 1
  - Controler 2
  - Country 1
  - Document 1
  - Pays 1
  - Projet 1
  - Rang 1
  - Receiver 1
  - Sender 1
  - TabCaveat 1
  - TabProjet 1
  - TabSecret
  - Table 1
  - User 1
  - Class Diagram 1




As Diagram

ToDo Item | Properties | Documentation | Style | Source | Constraints | Tagged Values | Checklist

Exec Node: 10

Feat State: Controler | c | Granted, AskAccessUser2, InitAS, initCT}

In Trans: Controler | c:c4

Out Trans: Controler | c:c10

OCL Constraints

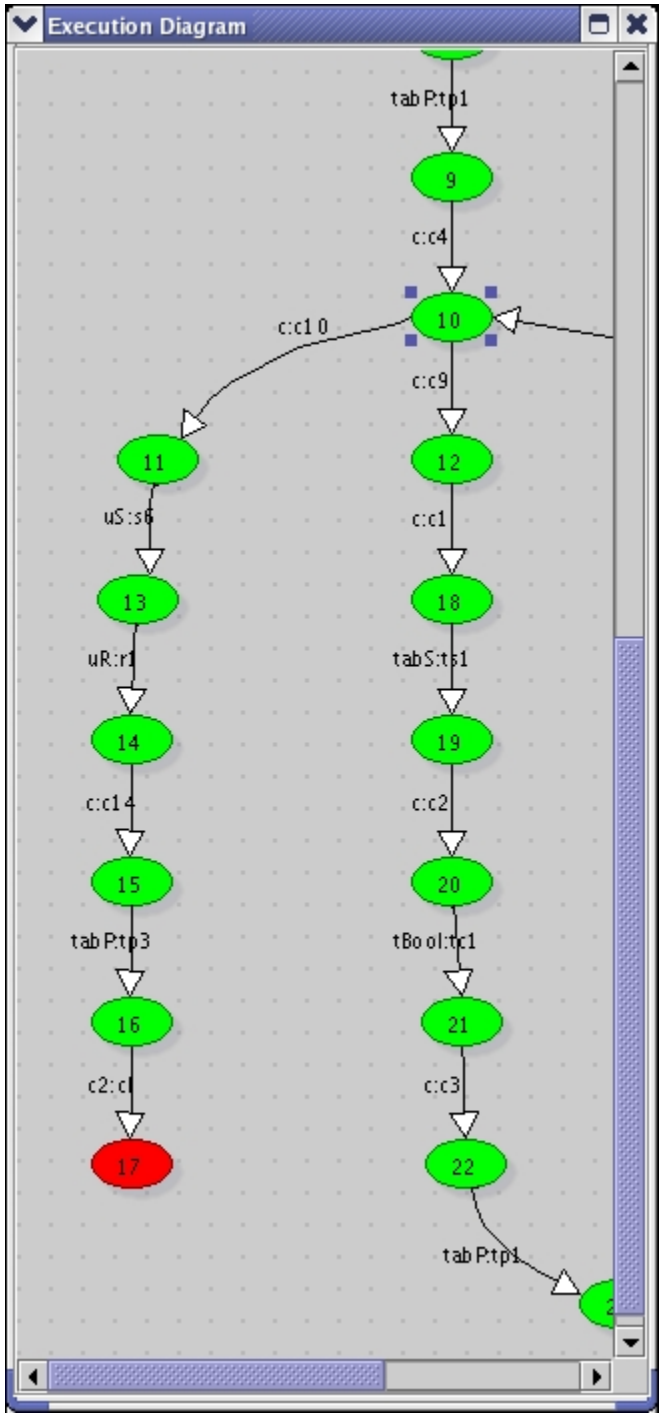
c_1_uR_1

Active States

d= {}
ger= {initC}
nato= {initCa}
tBool= {initTC}
tabP= {initTP}
tabS= {initSec}
uR= {initRec}
uS= {AskAccessSend, initSend1}
uk= {initC}
us= {initC}
usuk= {initCa}

Heap

uS.clearance=1
uS.con=[c]
uS.country=100
uS.doc=[d]
uS.rank=88
uS.sd=[uR]
uk.cav=[usuk,canusuk,nato]
uk.codeP=300
us.cav=[canus,usuk,canusuk,nato]
us.codeP=200
usuk.codeC=15


Execution Diagram
tabP:tp1
9
c:c4
10
c:c10
c:c9
11
12
uS:s6
c:c1
13
18
uR:r1
tabS:ts1
14
19
c:c14
c:c2
15
20
tabP:tp3
tBool:tc1
16
21
c2:c1
c:c3
17
22
tabP:tp1

# Plan of the Presentation

1. SOCLe Architecture
2. Caveat-Separation System: A Simplified Case Study
3. SOCLe Prototype
4. **Conclusion**

# Conclusion

- Critical software design should be and can be formally verified to some extent
- Reduces the risk of design problems and defects in the implementation
- Error detected
  - Falsified property when the access rules for documents can be updated while the point-to-point transfer is still in progress.
- Reasonably scalable
  - But state explosion on large systems
- Work in progress
  - State reduction techniques
  - On-the-fly validation

# Conclusion

- Technical reports
  - Caveat Case Study
  - An ASM Semantics of UML and OCL
  - Java Source Code Generation from UML
  - etc.
- Quebec Provincial FIQ OCTAS 2005 Winner
  - Finalist in the Canadian CIPA Awards
- IBM Canada at Ottawa is particularly interested in SOCLe technology transfer
  - Inline with their current vision
  - PolyMTL is currently preparing a proposition for IBM on how to integrate this technology into their tools

Frederic.Painchaud@drdc-rddc.gc.ca

http://www.polymtl.ca/crac/socle

DEFENCE R&D DÉFENSE