Some more steps on the way from APL+Win to Dyalog (Good and Bad Impressions)

Dr. Markos Mitsos
markos.mitsos@ergo.de

Deutsche Krankenversicherung AG DKV - ERGO, Actuarial Department

APL Germany — Bingen

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

Still valid:

- fight against everyday business
- keep operations (including "ad hoc") running while migrating

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

Still valid:

- fight against everyday business
- keep operations (including "ad hoc") running while migrating

About

Type of presentation:

- update on progress
- examples and demonstrations
- code management central

Still valid:

- fight against everyday business
- keep operations (including “ad hoc”) running while migrating

ERGO

Outline

1. Framework and structure

2. Deployment and examples

Outline

1. Framework and structure
2. Deployment and examples

Outline of section on framework and structure

In this section we outline:

Framework	code management and versioning
Structure	structure of WS `DIVERSES`

Outline of section on framework and structure

In this section we outline:

Framework code management and versioning

Structure structure of WS DIVERSES

ERGO

Outline of section on framework and structure

In this section we outline:

Framework	code management and versioning
Structure	structure of WS DIVERSES

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
    - no more copying between WS
    - no more shipping around of WS
- code management with Link
    - bi-directional link for coding
    - one-directional link for debugging
    - one-time import for testing
- versioning
    - in Tortoise SVN
    - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,...?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

ERGO

Checkout as master!

What is the framework for coding, debugging,. . . ?

- I bowed to the Inevitable
  - no more copying between WS
  - no more shipping around of WS
- code management with Link
  - bi-directional link for coding
  - one-directional link for debugging
  - one-time import for testing
- versioning
  - in Tortoise SVN
  - Git possible

▸ old config network ▸ new config network ▸ old config laptop ▸ new config laptop

ERGO

Top level namespaces

What is the top level structure of WS DIVERSES?

- namespace div with content proper
- namespace check with test cases
- namespace test with alternative algorithms, fooling around, notices, . . .
- reserved names (of namespaces) com, globals, dummy, . . .

Top level namespaces

What is the top level structure of WS `DIVERSES`?

- namespace `div` with content proper
- namespace `check` with test cases
- namespace `test` with alternative algorithms, fooling around, notices, . . .
- reserved names (of namespaces) `com`, `globals`, `dummy`, . . .

Top level namespaces

What is the top level structure of WS `DIVERSES`?

- namespace `div` with content proper
- namespace `check` with test cases
- namespace `test` with alternative algorithms, fooling around, notices, . . .
- reserved names (of namespaces) `com`, `globals`, `dummy`, . . .

Top level namespaces

What is the top level structure of WS `DIVERSES`?

- namespace `div` with content proper
- namespace `check` with test cases
- namespace `test` with alternative algorithms, fooling around, notices, . . .
- reserved names (of namespaces) `com`, `globals`, `dummy`, . . .

Top level namespaces

What is the top level structure of WS `DIVERSES`?

- namespace `div` with content proper
- namespace `check` with test cases
- namespace `test` with alternative algorithms, fooling around, notices, . . .
- reserved names (of namespaces) `com`, `globals`, `dummy`, . . .

Outline of section on deployment and examples

In this section we outline:

Deployment checking and "publishing" the workspace

Examples ADO, Excel, classes, key

Outline of section on deployment and examples

In this section we outline:

Deployment checking and "publishing" the workspace

Examples ADO, Excel, classes, key

Outline of section on deployment and examples

In this section we outline:

Deployment	checking and "publishing" the workspace
Examples	ADO, Excel, classes, key

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment (“publishing”)

Need medium to fix correct results and compare with actual ones.

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment ("publishing")

Need medium to fix correct results and compare with actual ones.

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment (“publishing”)

Need medium to fix correct results and compare with actual ones.

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment ("publishing")

Need medium to fix correct results and compare with actual ones.

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment (“publishing”)

Need medium to fix correct results and compare with actual ones.

ERGO

Simple, automated test

Automated testing of functional objects

- basic algorithms allow deterministic tests
- useful for development of WS itself
- also useful for new version of Dyalog
- automate tests and incorporate in deployment ("publishing")

Need medium to fix correct results and compare with actual ones.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

period system time for saving test results

Save test results in DB2:

- period system time very useful concept
- implicitly hidden also useful
- need also unicode support
- some problems with non-deterministic results and size to be solved

Comparisons can additionally be presented in Excel.

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Remaining specific problems with ADO and Excel

Many functions migrated, some problems remaining:

- asynchronous execution redirects results to Status Windows
- problems with events of ADO
- Excel not created as new instance
- behaviour different than under APL+Win (names with whitespace?)

Works in principle, but problems remain.

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Classes and key

Created some classes, working with key:

- some classes migrated including underlying Windows Objects
- problem with self destruction solved via global registration
- key operator key (!) for many basic algorithms
- in some cases use obviously successful, in others changes necessary

Some work done, some remaining.

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

ERGO

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

Conclusion

Stand of migration:

- strategy defined, framework build
- learned may things about Dyalog
- specific problems and requirements, especially front end
- pressure mounting, APL+Win 13 not getting newer!
- main applications still in feature, but proceeding with first ad hoc application

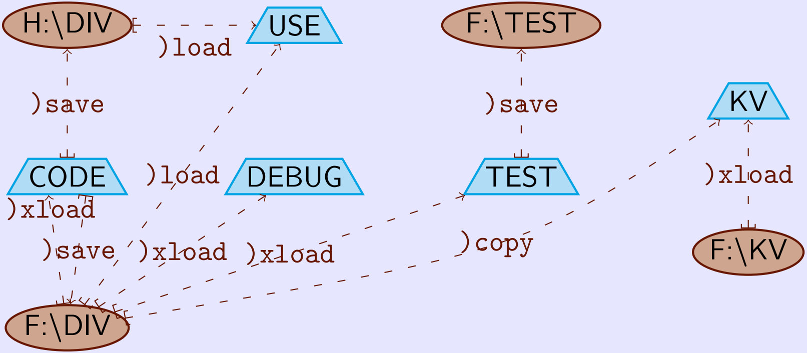
◂ begin

Overview of examples and illustrations

▸ old config network ▸ new config network ▸ old config laptop ▸ new config laptop

Old configuration in ERGO network

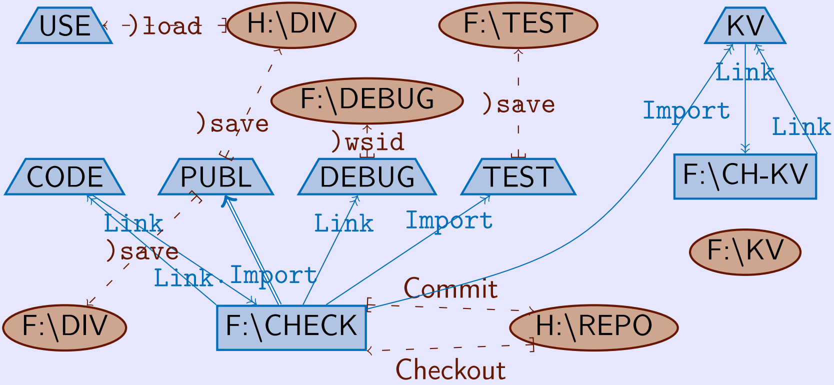
Working with WS as master:



Checkout as master!

Tentative new configuration in ERGO network

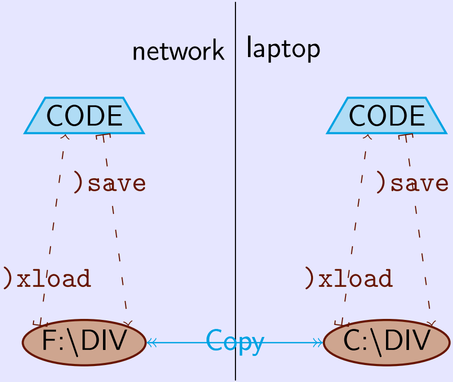
Working with checkout as master:



◂ Checkout as master!

Old configuration on laptop

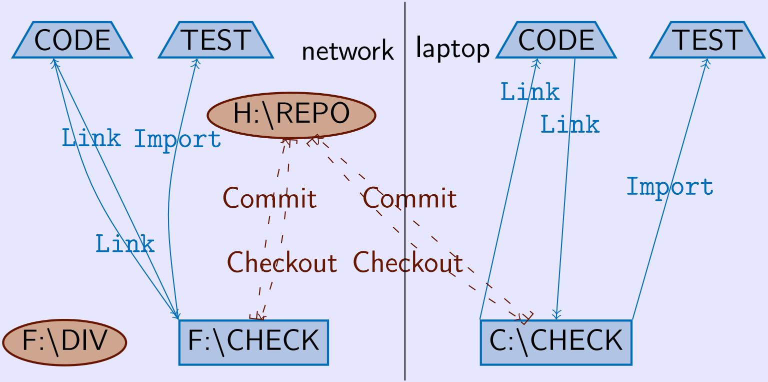
Working with WS as master:



◂ Checkout as master!

Tentative new configuration on laptop

Working with checkout as master:



◂ Checkout as master!