Current - Framework

```
try
{
        current_pkg->processEvent(Event);
}
catch (const std::exception& e) { … }
catch(const XRCP& e) { … }
```

Any exception causes the framework to stop execution. Text from the exception and context information are printed.

User Package (Observed)

```
Fwk::Result processEvent(edm::Event& e) {
        algorithm(e);
        return Result::success;
}

int algorithm(edm::Event& e) {
        … do stuff …
        if(error or bad) {
                return -1;
        else
                return 0;
}
```

Problem Area Example

```
Void func() {
        Clusters* x = new Clusters;
}
```

If new throw a “bad_alloc” exception,
it is caught by the framework because
not everyone catches this type of std::exception.
The exact location of the exception throw is
not known (core file stack information will not
contain it).

Not Catching Exceptions

- Stack may be intact, indicating where it was originally thrown.
- Other information (objects) in core file not reliable
- Algorithms should handle exceptions and abort the program if necessary to get a reliable core dump.
- With rethrows, stack will be where rethrow occurred

More About Exceptions

- The exception facility is not meant to be an excuse for not handling error in an algorithm.
- Blocks of code should be fitted with try/catch blocks, especially where resources are being acquired such as memory.
- Algorithm should throw exceptions when they cannot continue processing normally.

Destructors and Exceptions

```
struct thing { int value; }

void func1(int x) {
        thing t; <- destroyed before abort
        throw runtime_error(“bad”);
}
void func0(int x) {
        auto_ptr<thing> t(new thing); <- destroyed before abort
        func1(x);
}
int main() {
        auto_ptr<thing> t(new thing); <- destroyed before abort
        func0(10);
}
```

Proposal

- Introduce set of framework exceptions
  - AbortPackageError (skip only this package)
  - AbortGroupError (skip all in hook group)
  - AbortEventError (skip further processing)
  - AbortJob (stops execution completely)
- Exception definitions can exist is separate SRT package if necessary.
- Actual action taken by framework dictated by RCP settings.
- Matching set of return codes can be provided.

Framework RCP Example

Framework.rcp

string Packages="…"

string AbortPackageAction = "AbortJob"
string AbortGroupAction = "AbortGroup"
string AbortEventAction = "AbortEvent"
string AbortJobAction = "AbortJob"

Conclusion

- No work has been completed
- Does not solve the stack problem in core file
- Does not solve the problem of handling exception from the standard library