On the Future of std::future and a Future concept and Data-flow Programming

Hartmut Kaiser, Thomas Heller, Peter Sommerlad

2014-02-13

Document Number:	DXXXX
Date:	2014-02-13
Project:	Programming Language C++

1 History

1.1 Discussion on c++std-parallel mailing list

In 2013 there have been several discussions raised by papers **(find numbers)** that asked for extending `std::future` API with a member function `futre::then()` that allows to specify a function that will run after the future object becomes ready. The invocation of .then() would then return a future wrapping the original future object, etc.

Peter strongly objected to the abstraction of future gain "fat" by giving it more than the semantic of a *"ticket for a value or exception to be obtained later"*. While a concrete implementation such as std::future in the world of C++11 requires some hooking to a synchronization mechanism, the abstraction should be agnostic about where the value it eventually receives comes from.

Other seem to have the perspective that a `std::future` actually is about synchronization and thus chaining execution of code with respect to the event of a `std::future` instance becoming ready is the way to provide an attractive style of "continuation-based" programming **(check terminology)**.

2 Introduction

2.1

2.2 Open Issues to be Discussed

2.3 Acknowledgements

Acknowledgements go to

3 Proposed Library Additions