Asynchronous vs. synchronous programming

Is it worth to await?

Philip Wille
[1]Fakultät Informatik
Universität Innsbruck
September 7, 2020

Table of contents

Introduction

- A program interacts with many other systems, like databases, web services or file systems.
- A program that requests data from an external system must wait for the response.
- The time interval between sending a request and getting the corresponding response can take a long time.
- Without optimization the program is doing busy waiting and blocks therefore the main thread.
- This results in an unresponsive program. Therefore, for example a program with a graphical user interface (GUI) becomes unresponsive or can not provide a high throughput of requests.

Background

- Two main paradigms of programming: synchronous programming and asynchronous programming.
- Synchronous programming: Waiting for the termination of an action. The main thread is blocked in the meantime, because of busy waiting.
- Asynchronous programming: Waiting for an event that represents the termination of an action and notifies the program to proceeding in the work flow. The main thread can work on other tasks in the meantime.

Technical work

- Research question: Does benefit a server implementation from asynchronous programming?
- Trying to answer this question based on a proof of concept of CoAP.NET.
- CoAP.NET is an open source C# implementation of the **Co**nstrained **A**pplication **P**rotocol (CoAP).
- *"The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained (e.g., low-power, lossy) networks."* (Z. Schelby, June 2014)

Milestones

	Description
Milestone 1	Rewrite to .NET Standard 2.0
Milestone 2	Implement asynchronous pattern
Milestone 3	Specification from World-Direct
Milestone 4	Benchmark synchronous and asynchronous version
Milestone 5	Releasing V1.0