BINUS UNIVERSITY ONLINE LEARNING

People
Innovation
Excellence

LECTURE NOTES

Project Management

Topik 11

Advanced Topics in Planning and Scheduling

Agile and Critical Chain

People
Innovation
Excellence

LEARNING OUTCOMES

Setelah menyelesaikan bab ini, mahasiswa mampu :

- Memahami mengapa *Agile Project Management* dikembangkan dan kelebihannya dalam perencanaan untuk jenis proyek tertentu.
- Memahami fitur utama dari proses perencanaan Extreme Programming (XP) untuk proyek perangkat lunak.
- Memahami logika di balik Teori Kendala dan implikasinya untuk penjadwalan Rantai Kritis.
- Membedakan antara jalur kritis dan teknik penjadwalan proyek rantai kritis.
- Memahami bagaimana metodologi rantai kritis menyelesaikan konflik sumber daya proyek.
- Menjelaskan tentang kritik terhadap critical chain.

OUTLINE MATERI :

1. *Agile Project Management*
2. *Extreme Programming*
3. Teori Kendala dan Penjadwalan Proyek Rantai Kritis
4. Solusi *Critical Chain* dalam penjadwalan
5. *Critical chain* dalam konflik sumberdaya
6. Kritik terhadap *Critical Chain*

ISI MATERI

11.1. Agile Project Management

Salah satu metode manajemen proyek yang sangat populer adalah *Agile*. Dalam *Agile*, interaksi antar tim dan penyesuaian atas perubahan menjadi prioritas. Metode ini cocok digunakan untuk proyek yang bisa dipecah menjadi beberapa bagian. Bagian ini disebut dengan iterasi. Sejatinya, metode ini diciptakan untuk pengembangan *software*. Akan tetapi, seiring berjalannya waktu, metode ini bisa diterapkan untuk berbagai proyek lainnya.

11.1.1. Pengertian *Agile PM*

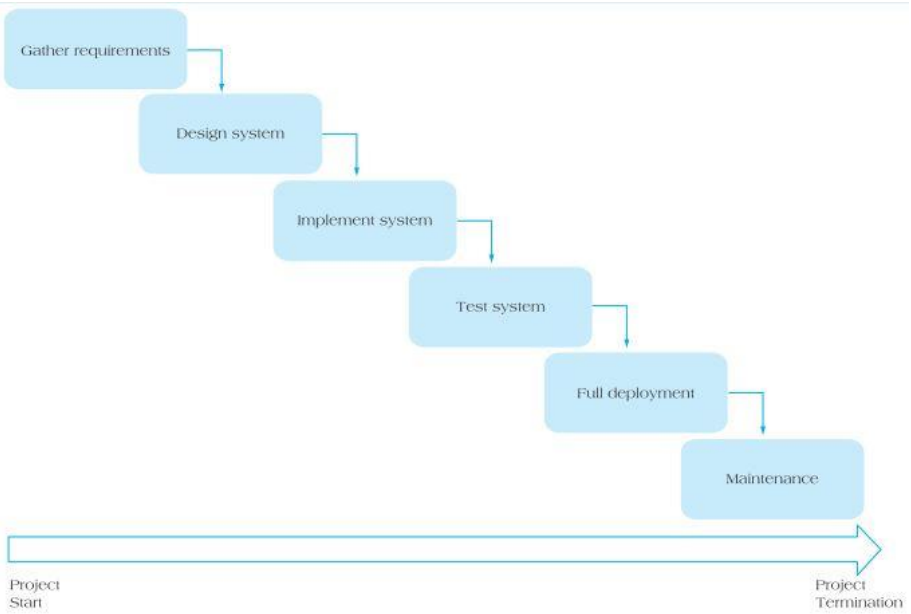
Agile Project Management (Agile PM) mencerminkan era baru dalam perencanaan proyek yang mengutamakan fleksibilitas dan kebutuhan pelanggan yang terus berkembang selama proses pengembangan. Agile PM berbeda dari manajemen proyek tradisional dalam beberapa hal, terutama melalui pengakuan bahwa pendekatan lama "merencanakan pekerjaan dan kemudian mengerjakan rencana" dengan hati-hati tidak mempertimbangkan kenyataan dari banyak proyek modern, yaitu bahwa kebutuhan pelanggan cenderung berkembang dan berubah selama proyek berlangsung. Mengikuti rencana awal yang masuk akal saat proyek dimulai mungkin tidak masuk akal saat proyek berjalan melalui tahap eksekusi. Agile PM menyadari pentingnya kebutuhan pelanggan yang terus berkembang ini dan memungkinkan proses perencanaan yang berulang dan bertahap—proses yang tetap terhubung dengan klien di seluruh siklus hidup proyek.

Agile PM menawarkan alternatif untuk proses perencanaan tradisional waterfall, yang menggunakan model siklus hidup sekuensial linier (lihat Gambar 11.1). Metode Waterfall membuat aliran pekerjaan secara linear, dari atas ke bawah, dari belakang ke depan. Misalnya, setiap tahap dalam proses pengembangan perangkat lunak terjadi secara berurutan, mengikuti penyelesaian tahap sebelumnya; persyaratan pertama dikumpulkan sepenuhnya, kemudian sistem dirancang, diimplementasikan, dan diuji. Akhirnya, itu sepenuhnya digunakan dan tunduk pada pemeliharaan. Dalam model waterfall, setiap fase harus diselesaikan sepenuhnya sebelum fase berikutnya dapat dimulai. Pada akhir setiap fase, dilakukan peninjauan untuk menentukan apakah proyek berada di jalur yang benar dan

apakah akan melanjutkan atau membuang proyek. Dalam model waterfall, fase tidak tumpang tindih. Proses pengembangan proyek waterfall bekerja dengan baik ketika:

- Persyaratan dipahami dengan sangat baik dan ditetapkan pada awal proyek.
- Definisi produk stabil dan tidak dapat berubah.
- Teknologi dipahami.
- Sumber daya yang cukup dengan keahlian yang dibutuhkan tersedia secara bebas.
- Proyek ini berdurasi relatif singkat.

Tetapi apa yang terjadi jika persyaratan berubah di tengah pengembangan proyek? Atau pelanggan memberikan serangkaian fitur "kritis" baru yang harus menjadi bagian dari produk akhir? Atau ketika inovasi teknologi baru memungkinkan tim kami untuk merampingkan perangkat lunak yang sedang kami kerjakan agar lebih ramah pengguna? Bagaimana jika asumsi awal atau ruang lingkup proyek dikomunikasikan dengan tidak baik, atau pesaing dapat mengalahkan dalam memasarkan produk yang identik? Dalam keadaan ini, keputusan untuk menyetujui ruang lingkup proyek yang sepenuhnya diartikulasikan yang ditetapkan di awal proyek dapat menyebabkan masalah kritis. Kekurangan metode waterfall, Adaptasi akan sulit dilakukan bilamana terjadi perubahan di tengah proyek. Dengan semakin banyak proyek yang mendukung inisiatif pengembangan produk baru yang memerlukan fleksibilitas untuk mengubah arah di tengah jalan, model waterfall dapat mengarah pada proses yang kaku yang tidak akan memberikan nilai kepada pelanggan, baik karena kebutuhan mereka terus berubah atau karena mereka melakukan kesalahan pekerjaan awalnya yang tidak sesuai dengan proyek yang seharusnya dilakukan. Untuk mengatasi masalah kritis inilah Agile PM dibuat.



Gambar 11.1. Model Waterfall dalam pengembangan proyek

11.1.2. Kelebihan Agile PM

Manajemen proyek tradisional menetapkan metodologi yang cukup kaku. Siklus hidup proyek menunjukkan bahwa konseptualisasi dan perencanaan dilakukan pada awal proyek. Konseptualisasi termasuk proyek membangun kasus bisnis (Mengapa kita melakukan ini? Apa yang ingin kita ciptakan? Bisakah kita menghasilkan keuntungan atau menciptakan nilai?) serta mengidentifikasi pemangku kepentingan utama proyek. Perencanaan berfokus pada pembuatan jadwal dan spesifikasi aktual untuk proyek. Dalam model manajemen proyek tradisional ini, kegiatan kritis ini diharapkan terjadi lebih awal dan ditangani secara komprehensif, tetapi setelah selesai dianggap selesai; dengan kata lain, sekarang saatnya untuk beralih ke pelaksanaan proyek. Mendasari pendekatan tradisional ini adalah asumsi bahwa anggota proyek dapat mengidentifikasi risiko (mengasumsikan ketidakpastian minimal) dan mengerjakan rencana yang telah mereka kembangkan sebelumnya (menganggap stabilitas maksimum).

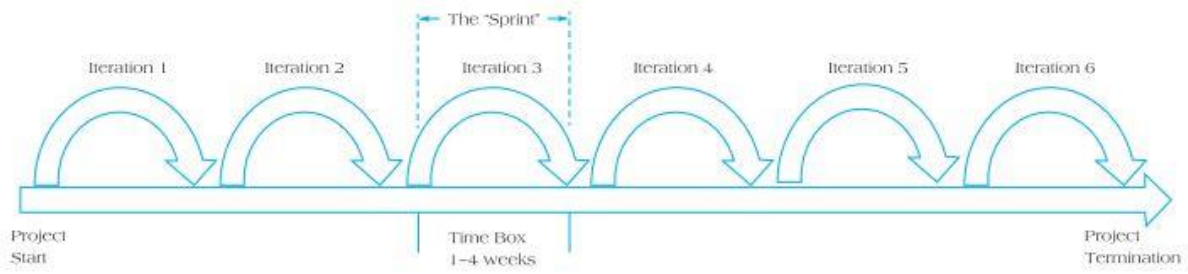
Untuk beberapa jenis proyek—jangka pendek, lingkup kecil, konstruksi, perencanaan acara, dan perbaikan proses—asumsi proyek tradisional ini tidak salah. Proyek dengan tujuan terbatas atau sempit atau yang akan selesai dengan cepat umumnya tidak mengalami masalah ketidakpastian yang signifikan. Jangka waktu yang singkat meminimalkan gangguan

lingkungan. Demikian juga, konstruksi, peningkatan produk saat ini, perencanaan acara, dan jenis proyek lainnya dengan tujuan yang jelas dan proses penyelesaian standar dapat direncanakan dengan cermat, biayanya diperkirakan secara akurat, dan jadwalnya dikembangkan secara wajar. Teknik perencanaan proyek tradisional bekerja dengan baik pada kelas proyek tertentu atau dalam situasi tertentu. Namun, semakin berada pada lingkungan yang tidak terduga, termasuk harus mempertimbangkan perubahan selera pelanggan, tantangan teknologi yang mengganggu, proses pengembangan yang kompleks, dan kerangka waktu yang lama, dengan menggunakan pendekatan perencanaan proyek tradisional, dimana asumsi stabilitas dan pengembangan dapat diprediksi, kemungkinan tidak akan berhasil.

Untuk mengatasi masalah pada perencanaan proyek tradisional, dikembangkan teknik baru yang inovatif yaitu teknik Agile. Agile PM, biasanya disebut sebagai *Scrum*, yang mengasumsikan bahwa setelah konseptualisasi dan perencanaan proyek awal selesai, proyek dapat dengan mudah dijalankan sesuai spesifikasi asli dan proyek yang diselesaikan akan menjadi "sukses". Misalnya, proyek perangkat lunak rentan terhadap perubahan. Ketika klien diharapkan dapat menyelesaikan semua persyaratan dan menunggu untuk waktu yang lama bahkan sebelum melihat prototipe, sangat tinggi kemungkinan si klien mengatakan, "Bukan itu yang saya inginkan!" Membuat rencana dan kemudian melepaskan diri dari pelanggan selama pengembangannya mungkin merupakan pendekatan tradisional untuk manajemen proyek, tetapi akan berbahaya pada proyek yang kompleks secara teknologi atau tidak pasti.

Agile PM adalah sistem yang fleksibel dan berulang yang dirancang untuk tantangan dalam mengelola proyek selama terjadi perubahan dan ketidakpastian yang konstan, sehingga ia memindahkan fungsi perencanaan dari lokasi awal tradisionalnya dalam siklus hidup proyek ke titik-titik di seluruh pengembangan proyek. Akibatnya, Agile PM menjadikan pengembangan sebagai proses "gelombang bergulir" dari siklus rencana-eksekusi-evaluasi yang berkelanjutan di seluruh pengembangan proyek (lihat Gambar 11.2). Tujuan dari setiap bagian di *Scrum* adalah untuk menciptakan nilai tambahan, melalui pengembangan subfitur atau elemen secara terus-menerus dalam keseluruhan proyek. Panjang iterasi pengembangan ini sengaja dibuat pendek—dari satu hingga empat minggu—cukup lama untuk membuat beberapa tambahan berharga pada proyek yang dapat dievaluasi pelanggan, tetapi cukup singkat untuk tetap berkomunikasi secara konstan dan menghadapi permintaan segera atau

modifikasi persyaratan. Setelah setiap siklus pengembangan, pertemuan tinjauan terjadi di mana fitur proyek dievaluasi, perubahan disetujui, spesifikasi dimodifikasi, dan hasil berikutnya diidentifikasi.



Gambar 11.2. Proses Scrum dalam pengembangan proyek

Untuk mengilustrasikan dengan contoh proyek perangkat lunak, Agile PM dapat mengurangi kompleksitas dan ketidakpastian yang ada pada pendekatan tradisional yang tidak efisien dan mahal dengan menghindari kesalahan umum dari proses yang berbulan-bulan dalam membuat persyaratan untuk keseluruhan proyek, menyelesaikan produk, dan kemudian mengujinya, untuk menemukan ratusan kekurangan dan fitur yang tidak dibutuhkan atau tidak berfungsi. Sebagai gantinya, subfitur atau bagian dari proyek perangkat lunak yang lebih kecil tetapi masih dapat digunakan akan diselesaikan satu per satu, kemudian diuji dan diverifikasi, semuanya dalam periode waktu yang lebih singkat. Dengan cara ini, setiap perubahan pada perangkat lunak tidak terlalu mahal baik dari segi waktu maupun uang.

Scrum berasal dari karya berkualitas Takeuchi dan Nonaka, yang mengenalkan metode holistik untuk pengembangan produk baru. Dalam model mereka, mereka berpendapat bahwa tim pengembangan harus bekerja sebagai unit yang terintegrasi untuk mencapai tujuan mereka. Pendekatan tradisional yang berurutan kadang-kadang disebut sebagai "keluar pagar" karena menggambarkan model di mana setiap kelompok fungsional menambahkan sesuatu ke produk dan kemudian, ketika selesai dengan upaya mereka, membawa proyek "keluar pagar" ke fungsi berikutnya pada kelompok yang diharapkan dapat melanjutkan proses pembangunan. Metode ini memperlambat pengembangan produk baru, menyebabkan kegagalan untuk menangkap fitur penting produk, mendorong miskomunikasi dan persaingan fungsional, dan akan meningkatkan biaya untuk memperbaiki produk di akhir siklus

pengembangan mereka, ketika kesalahan teknis dan kesalahpahaman fitur dapat menyebabkan proyek gagal dan uang yang terbuang.

11.1.3. Tugas Versus Cerita (*Story*)

Salah satu perbedaan kritis antara *Agile* dan perencanaan proyek tradisional berkaitan dengan sifat peran yang diasumsikan oleh anggota kunci. Dalam proses perencanaan proyek tradisional, sudut pandang pengembang proyek dianggap paling penting. Pengembang melihat proyek dari perspektif orang dalam; yaitu, berapa lama waktu yang dibutuhkan untuk pengembangan? Berapa banyak paket pekerjaan dan tugas yang diperlukan untuk menyelesaikan proyek? Pengembang proyek tertarik pada tugas karena mereka memungkinkan mereka untuk secara akurat dan efisien merencanakan pekerjaan dan membangun perkiraan biaya mereka. Semakin rinci dan spesifik tugas proyek, semakin mudah untuk membuat rencana dan perkiraan ini.

Cerita pengguna berbeda dan sangat penting untuk memahami kebutuhan nyata klien ("suara pelanggan"). Mereka ditulis oleh atau untuk pelanggan sebagai sarana untuk mempengaruhi pengembangan fungsionalitas produk. Semakin banyak pelanggan dapat menjelaskan apa yang mereka lakukan, apa yang mereka butuhkan, dan bagaimana mereka dapat menggunakan produk untuk melakukan pekerjaan mereka dengan lebih baik, semakin jelas cerita pengguna dan semakin baik kemampuan tim Scrum untuk mencapai tujuan ini. Cerita memiliki nilai karena mengidentifikasi pekerjaan aktual yang perlu dilakukan dengan produk atau sistem yang telah selesai. Setelah cerita divalidasi, mereka kemudian dapat didekonstruksi/ditata ulang menjadi tugas. Kuncinya terletak pada mengenali kebutuhan untuk mendengarkan cerita pengguna terlebih dahulu dan untuk mengidentifikasi nilai spesifik yang akan diberikan proyek. Misalnya, sebuah proyek yang mengabaikan suara pelanggan dapat dilakukan tepat waktu dan sesuai anggaran, tetapi tidak memberikan nilai nyata kepada pelanggan karena dilakukan tanpa mempertimbangkan terlebih dahulu untuk mengidentifikasi cerita pengguna yang kritis (apa yang mereka butuhkan untuk melakukan pekerjaan mereka lebih baik).

Karakteristik *Agile* lainnya yang menunjukkan pentingnya suara pelanggan adalah penekanan pada fitur produk, dibandingkan dengan menciptakan produk WBS yang mendetail. Jika asumsi yang digunakan dalam lingkungan Agile adalah karakteristik ruang lingkup proyek

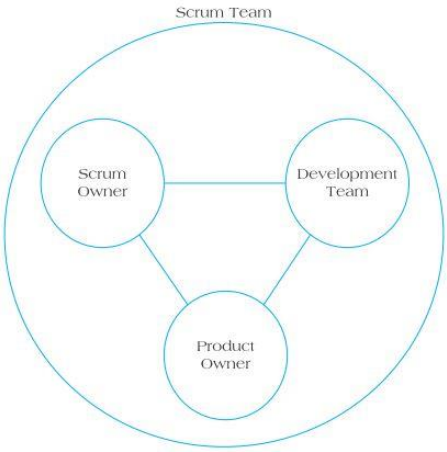
pasti akan berubah selama pengembangan proyek. Akibatnya, tidak masuk akal untuk menginvestasikan banyak waktu dan upaya untuk mengembangkan pernyataan ruang lingkup yang canggih untuk keseluruhan produk. Sebaliknya, Agile berfokus untuk mendapatkan fitur produk dengan benar: bagian dari produk yang memberikan nilai fungsional kepada pelanggan. Mendengarkan cerita pelanggan dan mendefinisikan hasil penting dari proyek dalam hal fitur akan memperkuat sifat kritis dari hubungan erat yang berkelanjutan yang harus dipertahankan oleh tim Scrum dengan klien.

11.1.4. Istilah Kunci pada PM Agile

Sprint—Sprint adalah salah satu iterasi dari siklus perencanaan dan pelaksanaan *Agile*. Sprint mewakili pekerjaan aktual yang sedang dilakukan pada beberapa komponen proyek dan harus diselesaikan sebelum pertemuan Scrum berikutnya.

Scrum—Dalam olahraga rugby, scrum adalah memulai kembali permainan setelah pelanggaran kecil. Scrum adalah suatu kerangka kerja untuk menangani masalah yang kompleks. Panduan bagi mereka yang ingin beradaptasi secara cepat dengan perubahan zaman, terutama dalam hal menyelesaikan masalah, membuat program, membuat layanan, mencari solusi secara cepat, kreatif dan inovatif. Metodologi scrum merupakan sebuah metode iteratif yang termasuk dalam metode Agile tentang bagaimana cara mengelola dan menjalankan sebuah proyek. *Scrum project management* terjadi ketika anggota tim memiliki beban tanggung jawab yang biasa dibebankan oleh *project manager*. Pemimpin dan fasilitator dalam pendekatan *scrum* dipegang oleh ***Scrum Master.***

Di Agile PM, Scrum mengacu pada strategi pengembangan yang disetujui oleh semua anggota kunci proyek (lihat Gambar 11.3). Pertemuan *scrum* melibatkan penilaian status proyek saat ini, mengevaluasi hasil *Sprint* sebelumnya, dan menetapkan tujuan dan kotak waktu untuk iterasi berikutnya.



Gambar 11.3. Scrum Team

Time-Box—Kotak waktu adalah panjang dari setiap sprint dan ditetapkan sebelumnya, selama rapat Scrum. Kotak waktu biasanya bervariasi antara satu sampai empat minggu.

User Stories (Cerita pengguna)—Penjelasan singkat dalam bahasa sehari-hari pengguna akhir yang menangkap apa yang mereka lakukan atau apa yang mereka butuhkan dari proyek yang sedang dikembangkan. Tujuan dari cerita pengguna adalah untuk mendapatkan perspektif pengguna tentang apa yang akan dilakukan produk yang dikembangkan dengan benar untuk mereka.

Scrum Master—Scrum Master adalah orang di tim proyek yang bertanggung jawab untuk memajukan proyek di antara iterasi, menghilangkan hambatan, dan menyelesaikan perbedaan pendapat di antara para pemangku kepentingan utama. *Scrum Master* tidak harus menjadi manajer proyek, tetapi memiliki peran formal dalam menegakkan aturan proses Scrum, termasuk memimpin rapat penting. Scrum Masters hanya berfokus pada proses pengembangan proyek Agile dan tidak berperan dalam manajemen sumber daya manusia.

Sprint Backlog—*Sprint Backlog* adalah kumpulan item Product Backlog yang dipilih untuk Sprint, ditambah rencana untuk mencapai Sprint Goal. Sprint Backlog adalah perkiraan oleh tim pengembangan tentang fungsionalitas apa yang akan ada dalam peningkatan kotak waktu berikutnya dan pekerjaan yang diperlukan untuk menyelesaikan fungsionalitas itu. Tim mengontrol Sprint Backlog.

Burndown Chart—*Sprint Burndown Chart* menunjukkan sisa pekerjaan di Sprint backlog.

Diperbarui setiap hari dan ditampilkan untuk dilihat semua anggota Scrum, ini memberikan referensi cepat tentang kemajuan Sprint.

Pemilik Produk—Orang yang mewakili pemangku kepentingan dan bertindak sebagai suara pelanggan." Pemilik produk dapat menjadi anggota organisasi proyek, tetapi harus mengambil sudut pandang pengguna dari luar dalam mewakili kebutuhan pelanggan. Pemilik produk membuat cerita pengguna yang mengidentifikasi kebutuhan spesifik mereka akan produk.

Tim Pengembang—Unit organisasi yang bertanggung jawab untuk mengirimkan produk pada akhir setiap iterasi (Sprint). Biasanya, tim pengembangan bersifat lintas fungsi dan mengatur diri sendiri; yaitu, mereka secara kolektif menentukan cara terbaik untuk mencapai tujuan mereka.

Product Backlog—Product Backlog adalah daftar prioritas dari segala sesuatu yang mungkin diperlukan dalam produk yang telah selesai dan merupakan sumber persyaratan untuk setiap perubahan yang akan dilakukan pada produk. Product Backlog tidak pernah final; melainkan berkembang seiring dengan berkembangnya produk dan lingkungan bisnis di mana ia akan digunakan. Itu terus berubah untuk mengidentifikasi apa yang dibutuhkan produk agar sesuai, kompetitif, dan bermanfaat. Pemilik produk mengontrol Product Backlog.

Work Backlog—Antrian bisnis dan fungsionalitas teknis yang berkembang dan diprioritaskan yang perlu dikembangkan menjadi sebuah sistem.

11.1.5. Langkah-Langkah *Agile*

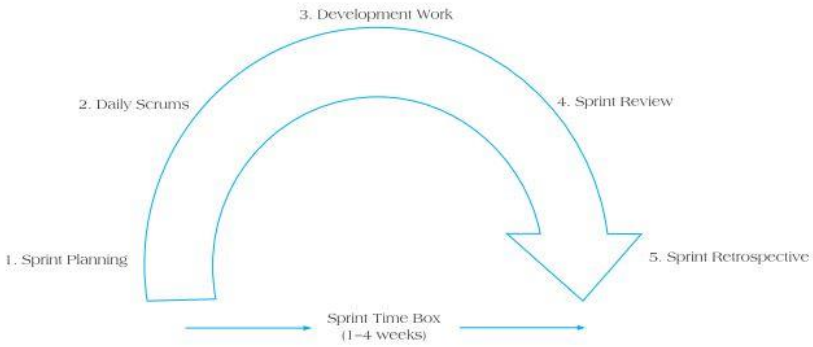
Proses *Agile* mengikuti serangkaian langkah yang memungkinkan metodologi untuk menggabungkan fleksibilitas yang dibutuhkan untuk menanggapi kebutuhan pelanggan dengan proses formal yang menciptakan urutan logis dalam menggunakan perencanaan *Agile*. Proses rapat singkat (*Scrum*) melibatkan serangkaian pertemuan yang mengelola proses pengembangan proyek melalui :

(1) Rapat Perencanaan cepat (*Sprint Planning*)
(2) *Scrum* Harian (*Daily Scrums*)
(3) Pekerjaan Pengembangan,
(4) Pertemuan *Review* Cepat, dan

(5) Rapat *Retrospective* (lihat Gambar 11.4)

Selama pertemuan (rapat), terdapat tiga pedoman untuk membentuk proses:

1. Tidak ada perubahan yang akan membahayakan atau mengubah tujuan Sprint. Setelah tujuan untuk Sprint disetujui, mereka tidak boleh diubah di tengah Sprint.
2. Sasaran mutu tidak menurun. Selama Sprint, tim tidak dapat mengubah tujuan atau mengorbankan standar kualitas yang telah disepakati terlebih dahulu.
3. Ruang lingkup dapat diklarifikasi dan dinegosiasikan kembali antara pemilik produk dan tim pengembangan seiring dengan semakin banyak yang dipelajari. Saat tim menemukan masalah atau peluang teknis selama Sprint, masalah atau peluang tersebut diteruskan ke pemilik produk untuk dipertimbangkan apakah akan mengubah ruang lingkup proyek atau tidak.



Gambar 11.4. Tahapan Sprint

(1) RAPAT PERENCANAAN SPRINT

Rapat/Pertemuan Perencanaan Sprint adalah titik awal Scrum, yang merupakan pertemuan dimana seluruh tim Scrum berkumpul; Bekerja sama dengan *Product Owner* dan *Master Scrum*, tim memilih *story* dari *backlog* dan melakukan *brainstorm* secara bersama-sama. Berdasarkan percakapan, kelompok Scrum memutuskan kompleksitas *story* dan memutuskan mana yang harus masuk ke sprint.

Pekerjaan yang akan dilakukan dalam *Sprint* diidentifikasi selama tahap Perencanaan *Sprint*. Rencana ini dibuat oleh kerja kolaboratif seluruh tim *Scrum*. Perencanaan *Sprint* dibatasi oleh waktu hingga maksimal satu hari penuh (delapan jam) untuk *Sprint* satu bulan. Untuk *Sprint* yang lebih pendek, lebih sedikit waktu yang diperlukan untuk Perencanaan *Sprint*. *Scrum Master* memastikan bahwa acara berlangsung dan semua anggota tim *Scrum* memahami

tujuannya. Sesi Perencanaan *Sprint* memperkenalkan *Sprint Backlog* kepada tim pengembangan dan mengoordinasikan upaya mereka untuk mencapai *item Backlog*.

Perencanaan Sprint menjawab pertanyaan-pertanyaan berikut:

- Apa yang dapat disampaikan dalam *increment* (kotak waktu) yang dihasilkan dari *Sprint* mendatang?
- Bagaimana pekerjaan yang dibutuhkan untuk mencapai peningkatan tersebut?

(2) *SCRUM* HARIAN

Melalui siklus sprint, setiap hari tim scrum bertemu maksimal lima belas menit (biasanya di pagi hari). Ada 3 hal yang akan dibahas oleh setiap member tim, yaitu:

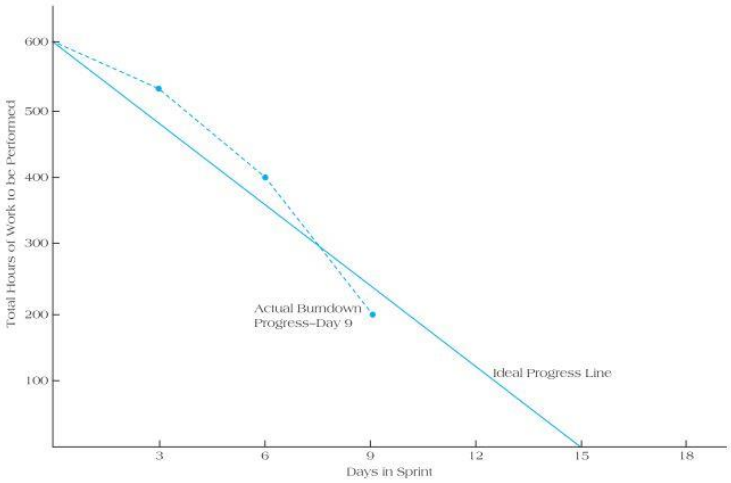
- Apa yang saya lakukan kemarin?
- Apa yang ingin saya lakukan hari ini?
- Menjelaskan jika orang tersebut memiliki sesuatu yang menghalangi mereka menyelesaikan pekerjaan mereka pada hari itu.

Sesi *Scrum* Harian semata-mata untuk tujuan informasi; hal ini hanya dimaksudkan untuk menjaga anggota tim dalam lingkaran komunikasi dan mengidentifikasi tren positif atau negatif yang mempengaruhi proyek. *Daily Scrums* juga menyertakan referensi ke *Burndown Chart*, yang merinci informasi terbaru tentang status item *Backlog* yang diselesaikan ("*burned down*") sejak pertemuan Scrum terakhir.

(3) PEKERJAAN PENGEMBANGAN

Pekerjaan Pengembangan adalah waktu ketika pekerjaan sebenarnya dari proyek sedang dilakukan selama *Sprint*. Kumpulan tujuan yang harus dicapai selama Sprint direpresentasikan pada *Burndown Chart* yang sedang berlangsung atau sudah selesai. Pekerjaan Pengembangan harus selalu dikoordinasikan antara anggota tim Scrum untuk memastikan bahwa tidak ada upaya yang sia-sia atau pekerjaan yang sedang dilakukan masuk pada item non-Sprint. Gambar 11.5 menunjukkan contoh *Burndown Chart* untuk Sprint dengan asumsi :

• Durasi sprint – 15 hari
• Ukuran tim – 5 anggota
• Jam/hari – 8
• Kapasitas total – 600 jam



Gambar 11.5 Contoh Grafik *Burndown*

(4) ULASAN SPRINT

Tinjauan *Sprint* diadakan di akhir *Sprint* untuk memeriksa peningkatan yang telah diselesaikan (*Sprint Backlog*) dan membuat perubahan pada *Product Backlog* jika diperlukan. Selama *Sprint Review*, tim *Scrum* dan pemangku kepentingan utama lainnya bekerja sama untuk memverifikasi apa yang telah dilakukan pada *Sprint*. Berdasarkan hasil ini dan setiap perubahan selanjutnya pada *Product Backlog*, tim *Scrum* sekarang merencanakan hal-hal berikutnya yang harus dilakukan untuk menambah nilai, termasuk fitur produk yang perlu diselesaikan atau dimodifikasi.

Ulasan Sprint adalah pertemuan informal. Hasil dari pertemuan *Sprint Review* adalah Product Backlog yang dimodifikasi dan ide dari item Sprint Backlog yang akan dibahas pada Sprint berikutnya.

Selama *Sprint Review*, kegiatan yang akan dilakukan meliputi:

- Pemilik produk menjelaskan item *Product Backlog* mana yang sudah selesai dan mana yang belum selesai.
- Tim pengembangan mendiskusikan apa yang berjalan dengan baik selama Sprint, masalah apa yang dihadapi, dan bagaimana masalah tersebut diselesaikan.

- Tim pengembang mendemonstrasikan pekerjaan yang telah diselesaikan dan menjawab pertanyaan tentang Sprint terbaru.
- Pemilik produk membahas *Product Backlog* sebagaimana adanya. Dia memproyeksikan kemungkinan tanggal penyelesaian berdasarkan kemajuan hingga saat ini jika diperlukan.
- Seluruh kelompok berkolaborasi tentang apa yang harus dilakukan selanjutnya, sehingga Tinjauan Sprint memberikan masukan yang berguna untuk Perencanaan Sprint berikutnya.
- Tinjau bagaimana pasar atau potensi penggunaan produk mungkin telah mengubah hal paling berharga berikutnya untuk diselesaikan.
- Tinjau garis waktu, anggaran, kemampuan potensial, dan pasar untuk rilis produk yang diantisipasi berikutnya.

(5) SPRINT RETROSPEKTIF

Pertemuan retrospektif terjadi setelah pertemuan review. Kelompok Scrum bertemu dan membicarakan hal-hal berikut:

- Apa saja yang terjadi dengan baik selama sprint
- Apa saja yang tidak berjalan seperti yang direncanakan dalam sprint.
- Pelajaran yang dipelajari
- Item tindakan yang harus ditindaklanjuti.

Scrum Master bekerja dengan tim Scrum untuk terus mengembangkan komunikasi dan mengidentifikasi perbaikan yang akan diperkenalkan pada Sprint berikutnya. Dengan cara ini, setiap Sprint tidak hanya membahas tentang bagaimana menyelesaikan pekerjaan (*Product Backlog*), tetapi juga digunakan untuk membangkitkan semangat untuk Sprint berikutnya sambil menciptakan tim yang lebih efisien dan produktif.

Dari pembahasan di atas dapat dilihat bahwa Scrum adalah solusi bagus untuk mendukung perkembangan proyek yang cepat dari hampir semua jenis proyek. Ini sangat efektif dalam menjamin efektifitas bagi organisasi manapun

11.1.6. Kunci Sukses Teknik *Agile*

Agar proses *Agile* menjadi paling efektif, diperlukan kemauan organisasi untuk mengadaptasi praktik dan filosofi operasi mereka dalam beberapa cara. Di antara kunci untuk membuat Agile bekerja adalah prinsip-prinsip berikut:

1. Tim lintas fungsi.

   Teknik Agile diakui paling kuat sebagai inisiatif berbasis tim. Mengembangkan tim yang terdiri dari orang-orang yang berkomitmen dari area fungsional kritis yang terpengaruh oleh pengembangan proyek yang sangat penting.

2. Memberdayakan anggota tim.

   Karena fokusnya adalah pada pengembangan hasil bisnis, tujuan metode Agile adalah membebaskan anggota tim untuk menjadi pemecah masalah; dengan kata lain, tidak terlalu khawatir tentang aktivitas mereka di proyek dan lebih memperhatikan nilai mereka sebagai pemecah masalah bersama untuk proyek tersebut.

3. Tanggung jawab bersama.

   Karena tim ditekankan di atas kontribusi individu, maka tidak seorang pun harus bertanggung jawab hanya untuk bagian proyeknya sendiri. Dengan cara ini, anggota tim termotivasi untuk berkomitmen pada keseluruhan proyek, bukan hanya bagian mereka sendiri dari upaya yang lebih besar.

4. Kepemimpinan yang melayani.

   Pemimpin proyek harus fokus pada menemukan cara untuk menyediakan sumber daya dan peluang bagi tim mereka untuk berhasil, daripada mengadopsi gaya manajemen mikro yang invasif. Peran kepemimpinan adalah untuk melayani tim, melalui penyediaan visi, sumber daya, atau dukungan kritis dalam menghadapi keterlibatan manajemen senior.

5. Aliran nilai yang berkelanjutan.

   Tujuan teknik Agile adalah untuk membuat garis waktu—melalui scrum, sprint, dan ulasan—yang berfokus pada penyediaan nilai yang stabil di seluruh siklus pengembangan proyek. Mengkoordinasikan output dari semua anggota tim di seluruh pengembangan memastikan bahwa nilai terus ditambahkan.

6. Perhatian terhadap keunggulan teknis.

   Memberikan nilai berarti bahwa kita memberikan nilai yang diharapkan pelanggan. Salah satu harapan penting adalah bahwa proyek tersebut memiliki kualitas teknis yang tinggi.

7. Pengurangan risiko secara cepat.

   Selain berfokus pada nilai bisnis, tim Agile perlu memahami bahwa perhatian penting lainnya dari pelanggan adalah risiko minimal yang terkait dengan proyek. Saat siklus pengembangan bergerak maju, maka sangat penting bahwa risiko ditangani dan (sejauh mungkin) dihilangkan pada tahap awal proses.

8. Umpan balik dan adaptasi awal.

   Kecepatan tidak masalah jika tim berkonsentrasi untuk menciptakan hasil proyek yang salah. Salah satu fitur penting dari Teknik Agile yang efektif adalah adanya adaptasi, pembentukan kembali, dan reorientasi tujuan proyek sepanjang siklus pengembangannya adalah apa yang mungkin diperlukan dalam mencapai kesuksesan akhir.

9. Keterbukaan dan transparansi total.

   Agile mengharuskan pelanggan, sejauh mungkin, menjadi anggota tim pengembangan yang setara. Menyembunyikan masalah atau kesalahan teknis dengan harapan bahwa pelanggan tidak akan menyadarinya sampai setelah penandatanganan proyek mengabaikan filosofi yang mendasari teknik tersebut. Tanpa keterbukaan, Teknik Agile tidak lebih baik dari teknik Waterfall.

10. Kepercayaan (*Trust*).

    Kepercayaan di antara semua pemangku kepentingan proyek adalah keuntungan yang kuat untuk pengembangan proyek Agile. Jika kepemimpinan yang melayani, keterbukaan, dan anggota tim yang diberdayakan berarti, pertama-tama harus ada kepercayaan mendasar di antara semua pemangku kepentingan.

Ketika teknik Agile dilakukan dengan benar, maka akan menjadi metode yang ampuh bagi organisasi untuk mengelola siklus pengembangan proyek.

11.1.7. Kelemahan Teknik Agile

Teknik Agile memberikan banyak keuntungan bagi perencana dan pengembang proyek, terutama dalam kelas proyek yang tujuan dari Agile paling relevan seperti manajemen proyek

TI atau pengembangan produk baru. Namun, beberapa kelemahan metodologi Agile harus dipertimbangkan juga.

Kelemahan ini meliputi:

1. Keterlibatan pengguna secara aktif dan kolaborasi erat dari tim Scrum sangat penting selama siklus pengembangan. Persyaratan ini sangat menuntut waktu dari semua pihak yang terlibat, dan mengharuskan pengguna untuk berkomitmen pada proses dari awal hingga akhir.
2. Persyaratan yang berkembang dapat menyebabkan perluasan cakupan di seluruh pengembangan, karena opsi atau perubahan baru selama Sprint dapat mengakibatkan serangkaian perubahan yang diminta oleh pengguna yang tidak pernah berakhir.
3. Karena persyaratan yang muncul dan fleksibilitas untuk membuat perubahan di tengah jalan, lebih sulit untuk memprediksi di awal proyek seperti apa produk akhir sebenarnya. Hal ini dapat menyulitkan untuk membuat kasus bisnis untuk proyek selama fase konseptualisasi dan menegosiasikan kontrak harga tetap dengan pelanggan atau vendor.
4. Persyaratan Agile dijaga seminimal mungkin, yang dapat menyebabkan kebingungan tentang hasil akhir. Dengan fleksibilitas untuk memperjelas atau mengubah persyaratan selama pengembangan proyek, ada sedikit informasi yang tersedia untuk anggota tim dan pengguna tentang fitur proyek dan cara kerjanya.
5. Pengujian terintegrasi di seluruh siklus hidup, yang menambah biaya proyek karena layanan personel teknis seperti penguji diperlukan selama keseluruhan proyek, bukan hanya di akhir.
6. Pengiriman fitur proyek yang sering (*Sprint Backlog*) di seluruh jadwal pengiriman tambahan proyek berarti bahwa pengujian dan penandatanganan fitur proyek hampir terus menerus. Hal ini membebani pemilik produk untuk siap dan aktif ketika serangkaian fitur baru muncul dari siklus Sprint terbaru.
7. Jika salah diterapkan pada proyek yang beroperasi di bawah prediktabilitas tinggi atau proses pengembangan terstruktur, persyaratan Agile untuk rescoping atau input pengguna yang sering dapat menjadi pendekatan mahal yang manfaatnya tidak sesuai dengan biayanya.

11.2. Pemrograman Ekstrim/ *Extreme Programming* (XP)

Extreme Programming (XP) adalah sebuah pendekatan atau model pengembangan perangkat lunak yang mencoba menyederhanakan berbagai tahapan dalam proses pengembangan tersebut sehingga menjadi lebih adaptif dan fleksibel. XP Pertama kali diusulkan oleh Kent Beck dan Ward Cunningham pada tahun 1996. Asal mula XP digunakan karena pada saat itu permintaan dari customer yang sering berubah dengan cepat sehingga mengakibatkan putaran kehidupan metode pengembangan perangkat lunak tradisional menjadi lebih pendek dan tidak selaras dengan metode tradisional karena pada umumnya memerlukan desain yang luas dan itu mengakibatkan perubahan desain yang terjadi dan tentu saja memerlukan biaya yang lebih tinggi. Tujuan XP adalah meminimalisir biaya yang diperlukan jika ada perubahan dalam pengembangan perangkat lunak.

XP juga mengadopsi penekanan Agile pada pentingnya story pengguna sebagai sarana untuk memahami kebutuhan nyata mereka, bukan interpretasi programmer dari pernyataan ruang lingkup yang kaku. XP mengambil namanya dari gagasan bahwa elemen inovatif dan bermanfaat dari praktik rekayasa perangkat lunak dibawa ke tingkat "ekstrim" dengan pendekatan ini.

Dua dari fitur pemandu XP adalah proses *refactoring* dan *pair programming*. Untuk mempercepat pengembangan perangkat lunak, pengujian fungsional dari semua persyaratan dilakukan sebelum pengkodean dimulai dan pengujian kode otomatis dilakukan terus menerus selama proyek berlangsung. Dalam XP, desain dijadikan kebutuhan intermediate. Desain dibuat sesederhana mungkin agar mudah mengimplementasikan code. Disini dapat terjadi perubahan struktur desain atau perubahan source code tanpa mengubah fungsi utamanya *(refactoring).* Fitur lain dari XP adalah pemrograman berpasangan (Pair programming). Pemrograman berpasangan adalah melakukan proses menulis program dengan berpasangan. Dua orang programer saling bekerjasama di komputer yang sama untuk menyelesaikan sebuah unit, di mana seorang programer menjadi "pengemudi" kodenya dan programer satunya menjadi "navigator" kode tersebut. Dengan melakukan ini maka keduanya selalu dapat berdiskusi dan saling melakukan koreksi apabila ada kesalahan dalam penulisan program. Aspek ini mungkin akan sulit dijalankan oleh para programer yang memiliki ego tinggi dan sering tidak nyaman untuk berbagi komputer bersama rekannnya. Pemrograman berpasangan dapat membantu pemrogram menyelesaikan masalah dan memperjelas

interpretasi cerita pengguna yang dapat memenuhi persyaratan. XP juga membutuhkan komunikasi yang konstan antara pelanggan dan tim pengembang. Bahkan, disarankan agar tim proyek tidak boleh berjumlah lebih dari 12 pengembang yang bekerja berpasangan.

Elemen lain dari *Extreme Programming* termasuk menghindari pemrograman fitur sampai benar-benar dibutuhkan, struktur manajemen yang datar, kesederhanaan dan kejelasan dalam kode, mengharapkan perubahan dalam persyaratan pelanggan seiring berjalannya waktu dan masalahnya lebih dipahami, dan sering berkomunikasi dengan pelanggan dan antar programmer. Sebagai pencetus Kent Beck mencatat: "Keuntungan dasar XP adalah seluruh proses terlihat dan akuntabel. Pengembang akan membuat komitmen nyata tentang apa yang akan mereka capai, menunjukkan kemajuan nyata dalam bentuk perangkat lunak yang dapat digunakan, dan ketika tonggak tercapai, mereka akan menjelaskan dengan tepat apa yang mereka lakukan dan bagaimana dan mengapa hal itu berbeda dari rencana. Hal ini memungkinkan orang yang berorientasi bisnis untuk membuat komitmen bisnis mereka sendiri dengan percaya diri, memanfaatkan peluang yang muncul, dan menghilangkan jalan buntu dengan cepat dan murah."

Agile PM dan XP telah berkembang dari kebutuhan untuk menggabungkan disiplin metodologi manajemen proyek dengan kebutuhan perusahaan modern untuk merespon peluang dengan cepat, mempromosikan lingkungan operasi internal komunikasi dan kolaborasi, dan tetap terhubung dan berkomitmen kepada pelanggan. Bagi beberapa jenis proyek, Agile PM menawarkan sarana untuk merampingkan proses pengembangan proyek sambil meningkatkan hasil *bottom line*. Dengan mengurangi biaya pengerjaan ulang dari kesalahpahaman atau perubahan persyaratan pelanggan, Agile PM telah ditunjukkan untuk menghemat uang organisasi proyek sambil meningkatkan hubungan pelanggan dan mendorong kelompok fungsional dalam organisasi untuk bekerja sama secara kooperatif. Secara keseluruhan, filosofi perencanaan Agile menawarkan banyak keuntungan bagi organisasi yang mengembangkan produk baru dengan cepat dan hemat biaya.

11.3. Teori Kendala dan Penjadwalan Proyek Rantai Kritis

11.3.1. Teori Kendala (*Theory of Constraints*-TOC)

Teori Kendala atau *Theory of Constraints* (TOC) merupakan suatu metode perubahan organisasi yang terfokus pada peningkatan laba. Teori ini adalah suatu filosofi manajemen

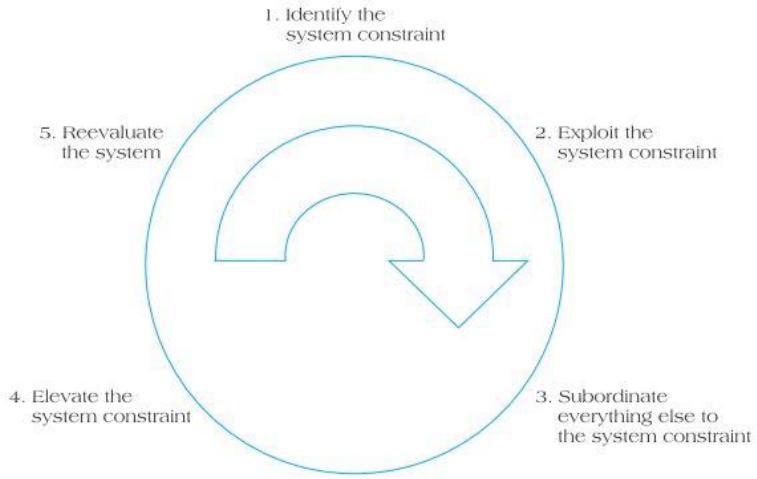
yang diperkenalkan pertama kali oleh Goldratt dalam bukunya yang berjudul “The Goal” pada tahun 1984. Sejak saat itu, TOC terus berevolusi dan berkembang dan saat ini menjadi faktor yang signifikan dalam dunia praktek manajemen.

Konsep penting dari TOC adalah bahwa setiap organisasi harus memiliki paling tidak satu kendala. Sebuah kendala merupakan suatu faktor yang membatasi organisasi untuk mendapatkan yang lebih dari usahanya, yaitu keuntungan. Tujuan ini berfokus pada kendala sebagai hambatan dari suatu proses dalam organisasi manufaktur. Namun, ada juga beberapa kendala non manufaktur, seperti permintaan pasar, atau kemampuan divisi sales untuk menerjemahkan permintaan pasar menjadi suatu order.

11.3.2. Lima Tahap Dasar TOC

Teory of Constraints memberikan metode spesifik untuk mengidentifikasi dan menghilangkan kendala-kendala yang ada, yang dikenal dengan *The Five Focusing Steps* atau 5 Langkah Dasar. Kelima langkah tersebut yaitu: (Gambar 11.6)

1. **Mengidentifikasi Sistem Kendala**, merupakan bagian dari sistem yang paling lemah, bisa berupa kendala fisik atau kebijakan.
2. **Memutuskan Bagaimana Mengeksploitasi Kendala**, yaitu melakukan perbaikan cepat ke seluruh kendala dengan memanfaatkan sumber daya yang ada.
3. **Subordinasi dan Sinkronisasi Kendala**, yaitu melakukan tinjauan terhadap semua kegiatan lain dalam proses untuk memastikan bahwa ada keselarasan.
4. **Meningkatkan Kinerja Kendala**, berupa pertimbangan mengenai tindakan lanjutan yang harus dilakukan apabila kendala masih tetap ada.
5. **Menghilangkan Kendala dan Melakukan Evaluasi Ulang terhadap Prosesnya**. Langkah ini berupa pengingat untuk terus memperbaiki kendala yang ada dan kemudian segera beralih pada kendala berikutnya.



Gambar 11.6. Lima Langkah Dasar TOC

Saat memeriksa jadwal proyek dari perspektif metodologi TOC, difokuskan pada kendala sistem utama, yaitu, satu akar penyebab dari mana semua masalah penjadwalan lainnya berkembang. Batasan sistem untuk proyek awalnya dianggap sebagai jalur kritis. Ingat, jalur kritis didefinisikan sebagai waktu sedini mungkin pada jaringan aktivitas yang diperlukan untuk menyelesaikan sebuah proyek. Jika kegiatan di jalur kritis tertunda, efeknya adalah menyebabkan keterlambatan proyek secara keseluruhan. Jalur kritis ditentukan oleh rangkaian kegiatan yang durasinya menentukan jalur terpanjang melalui jaringan dan oleh karena itu mengidentifikasi kemungkinan penyelesaian proyek sedini mungkin.

Goldratt mencatat bahwa semua penjadwalan dan masalah sumber daya yang terkait dengan proyek biasanya terjadi karena masalah dengan mencoba mempertahankan jalur kritis; karenanya, ini sering diidentifikasi sebagai kendala sistem utama.

11.4. Solusi Rantai kritis (*Critical Chain*) Penjadwalan Proyek

Critical Chain Project Management (CCPM) merupakan metode penjadwalan proyek yang diperkenalkan oleh Goldratt pada tahun 1997. Metode *Critical Chain* adalah metode yang membuat seorang manager proyek dapat menambahkan waktu penyangga (*buffer*) di setiap kegiatan dalam proyek tersebut dikarenakan terbatasnya sumber daya dan ketidakpastian proyek. Metode ini adalah perkembangan dari *Critical Path Method* dan mempertimbangkan efek akibat pemindahan sumber daya, optimalisasi sumber daya, pembagian sumber daya dan durasi dari kegiatan dari jalur kritis yang ditentukan dengan *Critical Path Method*. Untuk

mewujudkannya, *Critical Chain Project Management* (CCPM) menggunakan konsep *buffer* dan *buffer management*. CCPM menggunakan kegiatan yang di dalamnya tidak dimasukan *safety time* atau waktu aman tetapi menggantinya dengan *buffer time* atau waktu cadangan. *Buffer Time* terdiri dari *feeding buffer* dan *project buffer*. *Feeding buffer* adalah waktu penyangga yang menghubungkan aktivitas non-critical dengan aktivitas critical. Fungsinya adalah sebagai waktu cadangan untuk jika terdapat keterlambatan pada aktivitas non-kritis. *Project Buffer* adalah waktu cadangan yang diletakan pada akhir dari seluruh kegiatan proyek sebagai cadangan waktu keseluruhan proyek.

Solusi Goldratt untuk variabel yang terlibat dalam penjadwalan proyek melibatkan agregasi, atau kolektivisasi, dari semua risiko proyek dalam bentuk perkiraan durasi yang tidak pasti dan waktu penyelesaian.

Agregasi risiko adalah fenomena yang terkenal dalam bisnis asuransi. Teorema limit pusat menyatakan bahwa jika sejumlah distribusi probabilitas dijumlahkan, varians jumlah sama dengan jumlah varians dari distribusi individu. Rumus ini diberikan, di mana ada n distribusi independen dari varians yang sama V, sebagai:

$$V_{\Sigma} = n \times V$$

di mana V_{Σ} adalah varians dari jumlah. Standar deviasi σ dapat digunakan sebagai pengganti risiko, dan karena $\sigma^2 = V$, maka:

$$\sigma_{\Sigma} = (n)^{1/2} \times \sigma$$

di mana σ_{Σ} adalah simpangan baku jumlah. Karena itu:

$$\sigma_{\Sigma} < n \times \sigma$$

Secara matematis, rumus di atas menggambarkan poin untuk menggabungkan risiko yang mengarah pada pengurangan risiko secara keseluruhan.

Prinsip agregasi risiko yang sama ini dapat diterapkan dengan cara yang sedikit berbeda dengan metodologi rantai kritis. Di sini digunakan istilah *safety* atau penyangga proyek (*project buffer*) untuk merujuk pada cadangan kontingensi untuk aktivitas individu yang ingin dipertahankan oleh manajer proyek. Ketika kita mengumpulkan risiko, cadangan ini berkurang secara dramatis sehingga semua durasi aktivitas realistis tetapi menantang.

Artinya, menetapkan perkiraan durasi berdasarkan kemungkinan 90% penyelesaian yang berhasil, semua durasi aktivitas diperkirakan pada tingkat 50%. Ketentuan untuk kontinjensi, dalam bentuk keselamatan proyek, dihapus dari aktivitas individu dan diterapkan pada tingkat proyek. Karena konsep agregasi, buffer total ini lebih kecil dari jumlah buffer aktivitas proyek individu. Dengan demikian, durasi proyek berkurang.

Kisah sukses *Apple Computer Corporation* dengan peningkatan tablet iPad-nya menggambarkan beberapa keuntungan yang dapat ditemukan dalam menggabungkan risiko. Apple membuat keputusan sadar dengan iPad untuk mensubkontrakkan sebagian besar komponen produk ke berbagai pemasok. Perusahaan memutuskan bahwa untuk merekayasa seluruh produk akan menjadi alternatif yang kompleks dan berisiko.

Sebaliknya, ia mengontrak beberapa pemasok yang telah menghasilkan teknologi yang telah terbukti. Keputusan untuk menggabungkan komponen produk ini dari sumber lain, daripada memproduksinya sendiri, menyebabkan siklus pengembangan yang jauh lebih cepat dan profitabilitas yang sangat meningkat.

Dua pertanyaan mendasar yang harus dijawab adalah: Berapa tepatnya durasi proyek dapat dikurangi? Berapa banyak buffer agregat yang cukup? Goldratt dkk tidak menganjurkan menghapus semua buffer proyek, tetapi hanya penerapan kembali buffer itu ke tingkat proyek (seperti yang ditunjukkan pada Gambar 11.7). Penentuan jumlah buffer yang tepat untuk dipertahankan dapat diperoleh dengan salah satu dari dua cara:

(1) pendekatan "*rule of thumb*" (aturan umum) yang disarankan Goldratt, yaitu, mempertahankan 50% dari total buffer proyek; dan

(2) model yang diturunkan secara matematis menurut Newbold (1998) adalah :

$$\text{Buffer} = \sigma = [((w_1 - a_1)/2)^2 + ((w_2 - a_2)/2)^2 + \ldots + ((w_i - a_i)/2)^2]^{1/2}$$

di mana w_i adalah durasi kasus terburuk dan a_i adalah durasi rata-rata untuk setiap tugas yang menyediakan bagian dari nilai buffer agregat. Standar deviasi yang diperkirakan adalah $(w_i - a_i /2)$. Misalkan, tim proyek mencari buffer yang panjangnya 2 kali standar deviasi. Rumus untuk menghitung panjang buffer yang sesuai adalah:

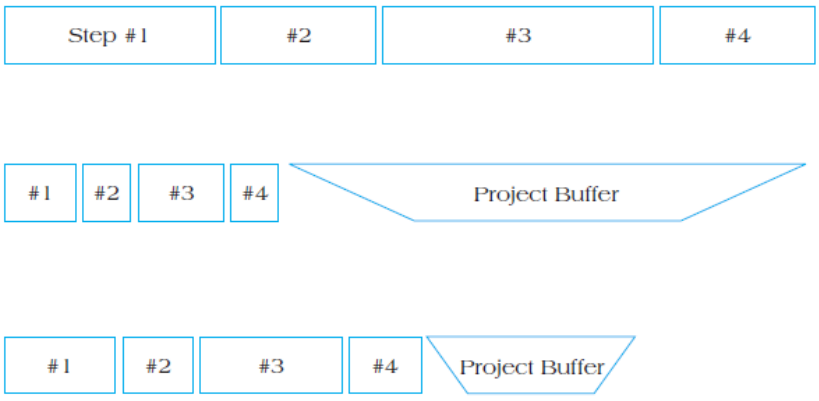
$$\text{Buffer} = 2 \times \sigma = 2 \times [((w_1 - a_1)/2)^2 + ((w_2 - a_2)/2)^2 + \ldots + ((w_i - a_i)/2)^2]^{1/2}$$

Sebagai contoh kita asumsikan, misalnya, bahwa kita memiliki tiga tugas yang saling terkait, masing-masing berdurasi 20 hari. Jadi, kasus terburuk (w_i) untuk jangka waktu ini adalah 20 hari asli. Selanjutnya, dengan menggabungkan buffer berdasarkan solusi 50%, nilai i adalah 10 hari untuk setiap aktivitas. Kita dapat memperoleh ukuran buffer yang sesuai (dua standar deviasi) dengan:

Buffer = $\sqrt{((20_1 - 10_1)^2 + (20_2 - 10_2)^2 + (20_3 - 10_3)^2)} = \sqrt{300}$ = 17.32 hari

Secara visual, kita dapat memahami penerapan CCPM dalam tiga fase yang berbeda.

1. Langkah Pertama, semua tugas proyek yang relevan diletakkan dalam diagram prioritas yang disederhanakan (ditunjukkan pada baris 1 pada Gambar 11.7), dengan durasi yang diantisipasi ditentukan. Ingat bahwa perkiraan durasi asli kemungkinan besar didasarkan pada kemungkinan tinggi perkiraan penyelesaian dan oleh karena itu memerlukan pemeriksaan ulang berdasarkan penilaian yang lebih realistis dari durasi "sebenarnya".
2. Langkah kedua terdiri dari mengurangi perkiraan durasi ini ke tingkat kemungkinan 50%. Semua *safety* tugas individu, atau buffer, telah dikumpulkan dan sekarang diberikan sebagai penyangga/ buffer tingkat proyek.



Gambar 11.7. Pengurangan Durasi Proyek Setelah Agregasi

Pada tahap ini, panjang keseluruhan proyek tidak berubah karena buffer tugas individu hanya dikumpulkan dan ditambahkan ke akhir jadwal proyek. Namun, baris 3 mengilustrasikan langkah terakhir dalam konfigurasi ulang, titik di mana buffer proyek berkurang dengan jumlah yang dapat diidentifikasi. Menggunakan aturan praktis penyusutan 50%, kami berakhir dengan jadwal proyek yang masih jauh lebih pendek dari aslinya. Jadwal yang

diubah dan dipersingkat ini mencakup beberapa kelonggaran untuk setiap aktivitas. Akibatnya, CCPM menyebabkan jadwal proyek dipersingkat.

Misalkan diagram jaringan aktivitas proyek menghasilkan nilai awal yang diberikan pada Tabel 11.1.

Tabel 11.1. Critical Chain Activities Time Reductions

Activity	Original Estimated Duration	Duration Based on 50% Probability
A	10 days	5 days
B	6 days	2 days
C	14 days	7 days
D	2 days	1 day
E	8 days	3 days
Total	40 days	18 days

Perhatikan bahwa jaringan yang dimodifikasi memperpendek durasi proyek secara keseluruhan hingga 22 hari, dari semula 40 menjadi 18 hari. Karena semua risiko sekarang dikumpulkan di tingkat proyek, ada total 22 hari potensi slack dalam jadwal yang dihasilkan dari perkiraan aktivitas yang menyusut pada setiap tahapan proyek. Jadwal proyek yang dimodifikasi CCPM akan menerapkan kembali 11 hari dari penyusutan jadwal yang diperoleh yang difungsikan sebagai buffer proyek secara keseluruhan. Oleh karena itu, jadwal proyek baru akan mengantisipasi durasi yang diperkirakan membutuhkan 29 hari untuk diselesaikan.

Apa implikasi dari penerapan kembali slack proyek ini ke tingkat agregat? Pertama, semua tanggal jatuh tempo untuk aktivitas individu dan subaktivitas telah dieliminasi. Tonggak sejarah tidak digunakan dalam jaringan aktivitas CCPM. Satu-satunya komitmen tegas tetap pada tanggal pengiriman proyek, bukan pada penyelesaian tugas individu. Anggota tim proyek didorong untuk membuat perkiraan yang realistis dan terus-menerus dapat mengomunikasikan harapan mereka. Jelas, agar CCPM dapat bekerja, budaya perusahaan yang mendukung kebijakan “tidak menyalahkan” sangat penting. Ingat, sifat membutuhkan perkiraan kemungkinan 50% untuk durasi aktivitas individu menyiratkan bahwa pekerja kemungkinan besar akan melewatkan tanggal komitmen untuk mencapainya. Di bawah budaya yang secara rutin menghukum kinerja yang terlambat, pekerja akan dengan cepat

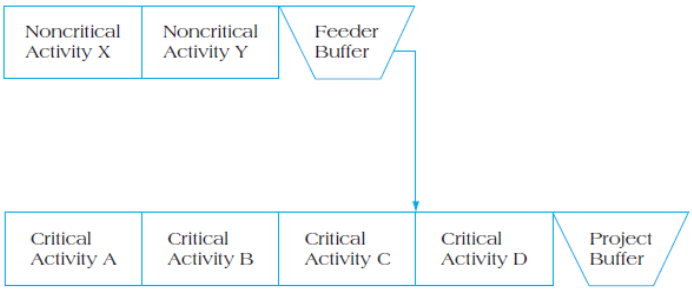
memperoleh kembali kebiasaan yang pernah mereka lakukan—perkiraan yang dibesar-besarkan, menyia-nyiakan keselamatan, dan sebagainya.

Implikasi kedua mungkin lebih signifikan, terutama ketika berurusan dengan sub-kontraktor eksternal. Karena tanggal aktivitas individu telah dihilangkan dan tonggak pencapaian dihapus, maka menjadi masalah untuk menjadwalkan pengiriman subkontraktor secara efektif. Ketika subkontraktor setuju untuk menyediakan bahan untuk proyek, mereka secara rutin beroperasi sesuai dengan tanggal pengiriman (kalender). CCPM, dengan filosofinya yang tidak menekankan tanggal target untuk tugas individu, menciptakan lingkungan yang rumit untuk menjadwalkan pengiriman pemasok atau subkontraktor yang diperlukan. Penulis CCPM menyarankan bahwa salah satu metode untuk mengurangi kekhawatiran ini adalah bekerja dengan kontraktor untuk menegosiasikan penyelesaian awal dan pengiriman komponen yang diperlukan untuk kegiatan kritis.

11.4.1. Mengembangkan Jaringan Aktivitas Rantai Kritis

Pada bab-bab sebelumnya dijelaskan bahwa dengan jaringan CPM/PERT tradisional, slack aktivitas individu merupakan artefak keseluruhan jaringan. Waktu mulai aktivitas biasanya ditentukan oleh ketersediaan sumber daya. Misalnya, meskipun suatu kegiatan dapat dimulai paling cepat pada tanggal 15 Mei, namun dapat menundanya selama tiga hari karena orang yang bertanggung jawab atas penyelesaiannya tidak tersedia hingga tanggal tersebut. Dengan cara ini, float digunakan sebagai perangkat pemerataan sumber daya.

Dengan CCPM, *leveling* sumber daya tidak diperlukan karena sumber daya diratakan dalam proyek dalam proses mengidentifikasi rantai kritis. Untuk penjadwalan, CCPM menganjurkan untuk menunda semua aktivitas nonkritis selambat mungkin, sambil menyediakan setiap jalur nonkritis dalam jaringan dengan buffernya sendiri (lihat Gambar 11.8). Buffer nonkritis ini disebut sebagai buffer pengumpan karena mereka ditempatkan di mana jalur nonkritis masuk ke jalur kritis. Seperti yang ditunjukkan Gambar 11.8, sebagian dari jalur kritis dan salah satu jalur pengumpan nonkritis bergabung hanya melewati titik aktivitas C. Durasi buffer pengumpanan dihitung mirip dengan proses yang digunakan untuk membuat buffer proyek keseluruhan, yang dilampirkan ke akhir rantai kritis.

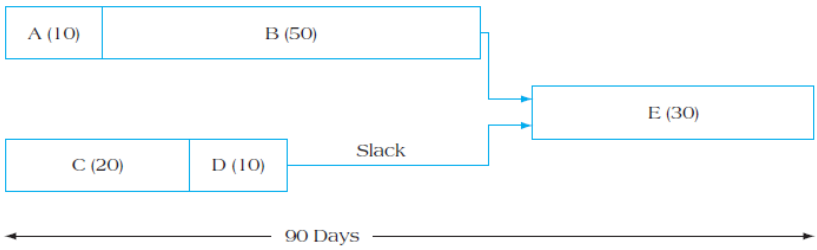


Gambar 11.8. CCPM dengan Buffer pengumpan

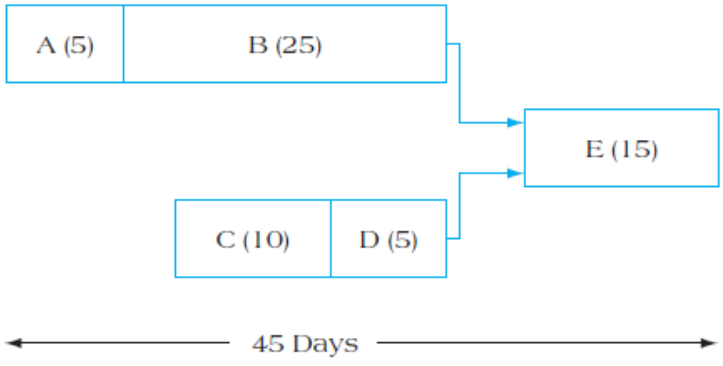
Untuk memahami bagaimana logika rantai kritis dibangun, perhatikan bahwa langkah pertama terletak pada membuat beberapa penyesuaian penting untuk pendekatan penjadwalan tradisional, seperti:

1. Menyesuaikan durasi aktivitas yang diharapkan untuk mencerminkan 50% kemungkinan penyelesaian tepat waktu (menyusut jadwal)
2. Berubah dari proses awal-awal ke pendekatan akhir-akhir
3. Memperhitungkan efek perebutan sumber daya jika perlu

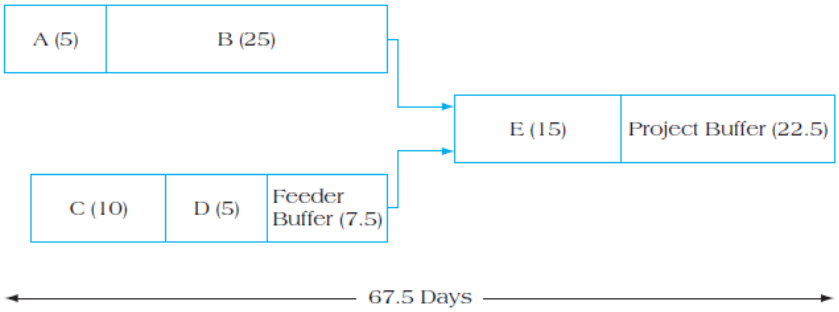
Gambar 11.9a, b, dan c menyajikan serangkaian contoh sederhana yang mengikuti langkah-langkah ini.



Gambar 11.9a. Project Schedule Using Early Start



Gambar 11.9b. Reduced Schedule Using Late Start



Gambar 11.9c. Critical Chain Schedule with Buffers Added

Tabel 11.2 Durasi aktivitas

Activitas	Durasi (hari)
A	10
B	50
C	20
D	10
E	30

Gambar 11.9a menunjukkan jaringan aktivitas standar berdasarkan pendekatan PERT. Sebanyak lima aktivitas diidentifikasi (A, B, C, D, dan E) di sepanjang dua jalur terpisah yang dimasukkan ke dalam aktivitas E pada akhir proyek. Semua kegiatan dijadwalkan untuk dimulai sedini mungkin (*early start*) dan didasarkan pada metode standar untuk memperkirakan durasi. Tabel 11.2 menunjukkan durasi yang diharapkan.

Gambar 11.9a menunjukkan perkiraan durasi proyek keseluruhan selama 90 hari, berdasarkan rangkaian aktivitas terkait terpanjang (jalur A – B – E). Jalur kedua, C – D – E, memiliki durasi keseluruhan 60 hari dan karenanya, memiliki slack selama 30 hari. Untuk menyesuaikan jaringan ini, langkah pertama melibatkan perubahan ke jadwal mulai terlambat. Kedua, CCPM memberikan perkiraan durasi aktivitas asli dan menggantikannya berdasarkan titik rata-rata distribusi. Jaringan aktivitas yang dimodifikasi mengasumsikan pengurangan perkiraan sebesar 50%. Oleh karena itu, jaringan baru memiliki durasi keseluruhan 45 hari, bukan perkiraan 90 hari yang asli (Gambar 11.9b).

Langkah selanjutnya dalam konversi ke jadwal rantai kritis melibatkan masuknya buffer proyek dan feeder untuk semua jalur jaringan. Ingat bahwa buffer ini dihitung berdasarkan penerapan 50% dari keseluruhan penghematan jadwal. Buffer feeder untuk jalur C – D dihitung sebagai (0.50)(10 + 5), atau 7,5 hari. Buffer proyek, ditemukan dari nilai untuk jalur A – B – E, dihitung sebagai (0.50)(5 + 25 + 15), atau 22,5 hari. Oleh karena itu, setelah buffer ditambahkan ke jaringan aktivitas yang dimodifikasi, bagan PERT asli menunjukkan durasi 90 hari dengan slack 30 hari, jaringan rantai kritis baru memiliki durasi keseluruhan 67,5 hari, atau penghematan 22,5 hari (Gambar 11.9c). Oleh karena itu, melalui tiga langkah, dapat beralih dari jadwal mulai awal ke jadwal mulai akhir, mengidentifikasi jalur kritis (urutan aktivitas terkait terpanjang), dan kemudian menerapkan feeder dan buffer proyek. Hasilnya adalah jadwal proyek yang dimodifikasi, bahkan dengan buffer dimasukkan, secara signifikan mengurangi waktu penyelesaian yang dijadwalkan untuk proyek tersebut.

11.4.2. Solusi Rantai Kritis Vs Solusi Jalur Kritis

Apa perbedaan nyata antara metode jalur kritis dan Manajemen Proyek Rantai Kritis? Rantai kritis biasanya bukan jalur yang sama dengan jalur kritis dalam jaringan aktivitas. Jalur kritis hanya bergantung pada ketergantungan tugas, yaitu hubungan tugas dengan pendahulunya.

Dalam proses ini, slack aktivitas ditemukan setelah fakta; setelah jaringan ditata dan jalur kritis diidentifikasi, semua jalur dan aktivitas lain mungkin mengandung beberapa tingkat slack. Di sisi lain, rantai kritis biasanya melewati ketergantungan tugas. Sekali lagi, efek ini terjadi karena rantai kritis mengharuskan semua leveling sumber daya dilakukan sebelum rantai kritis dapat diidentifikasi, bukan setelahnya seperti dalam kasus jaringan PERT dan CPM.

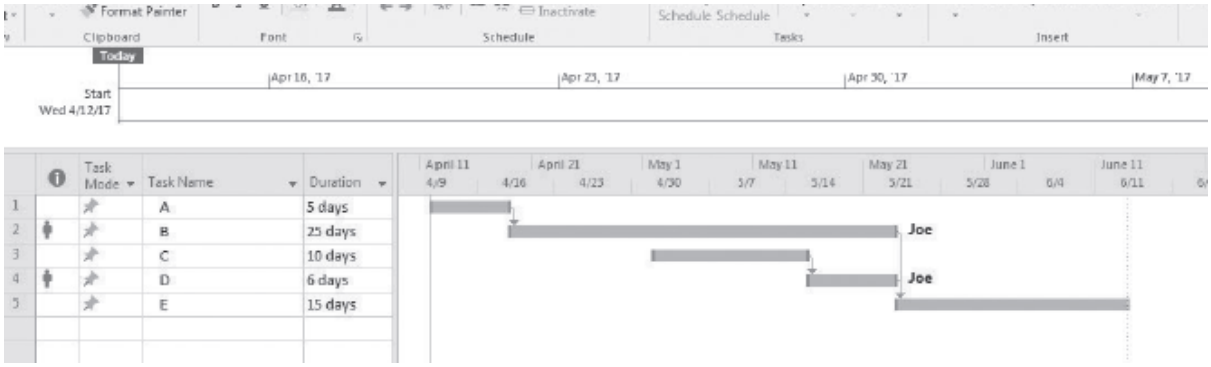
11.5. *Critical Chain* Dalam Konflik Sumberdaya

Misalkan setelah menyusun jadwal yang direvisi (lihat Gambar 11.9c), kita menemukan titik perebutan sumber daya. Kita asumsikan bahwa aktivitas B dan D membutuhkan orang yang sama, menghasilkan sumber daya yang kelebihan beban. Bagaimana cara kita mengatasi kesulitan tersebut? Karena tanggal mulai semua kegiatan diundur selambat-lambatnya, maka langkah-langkah yang harus dilakukan adalah sebagai berikut:

1. Tugas sebelumnya untuk aktivitas D adalah aktivitas C. Oleh karena itu, langkah pertama terletak pada penetapan batasan mulai kendala awal selambat mungkin ke aktivitas C.

2. Untuk menghilangkan konflik sumber daya, bekerja mundur dari akhir proyek, menghilangkan sumber konflik.

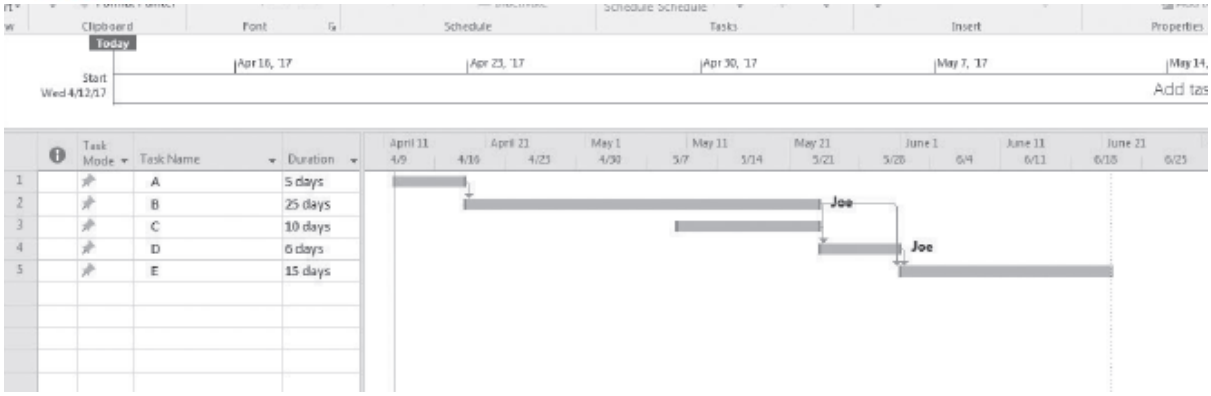
Gambar 11.10 menyajikan file Proyek MS yang menggambarkan langkah-langkah dalam menyesuaikan jadwal rantai kritis untuk menghilangkan konflik sumber daya. Perhatikan bahwa gambar asli (Gambar 11.9c) memperlihatkan masalah standar ketika pengembangan jadwal tipe mulai awal (*early-start*) yaitu, kebutuhan untuk mengevaluasi jadwal terhadap kemungkinan kelebihan sumber daya. Misalkan, Gantt Chart (Gambar 11.10) menunjukkan konflik sumber daya (Joe) yang ditugaskan baik aktivitas B dan D selama seminggu dari tanggal 6 Maret. Karena orang ini tidak dapat melakukan kedua aktivitas secara bersamaan, maka kita harus mengkonfigurasi ulang jadwal untuk memungkinkan kendala ini.



Gambar 11.10. Penjadwalan Menggunakan Late Start untuk Aktivitas Proyek

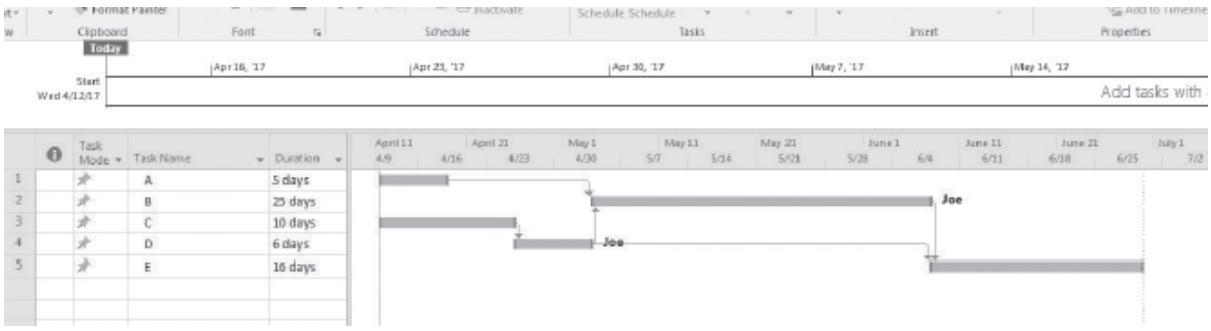
Sumber: MS Project 2016, Microsoft Corporation.

Gambar 11.11 menunjukkan langkah selanjutnya dalam proses penyelesaian konflik sumber daya. Sambil mempertahankan format terlambat memulai (*late start*), aktivitas D didorong kembali terjadi setelah aktivitas B, sehingga memungkinkan Joe untuk melakukan B terlebih dahulu sebelum pindah ke tugas berikutnya. Jumlah penundaan jadwal total sekitar satu minggu dengan jadwal yang dikonfigurasi ulang.



Gambar 11.11. Mengonfigurasi Ulang Jadwal Penyelesaian Konflik Sumber Daya

Atau, masalah konflik sumber daya ini dapat dijadwal ulang menurut Gambar 11.12, di mana aktivitas C dan D dipindahkan ke depan dalam jaringan. Solusi alternatif ini memang menambah waktu tambahan ke jalur jaringan, memindahkan tanggal penyelesaian yang diproyeksikan ke minggu kedua di bulan April. Saat memilih solusi yang paling tepat untuk masalah konflik sumber daya, Anda menginginkan opsi yang meminimalkan gangguan jadwal jaringan total. Dalam contoh yang ditunjukkan, mungkin lebih baik untuk mengadopsi jadwal yang ditunjukkan pada Gambar 11.11 karena mengatasi konflik sumber daya dan menawarkan jadwal yang dikonfigurasi ulang yang hanya kehilangan satu minggu secara keseluruhan.



Gambar 11.12. Solusi Alternatif pada Masalah Konflik Sumber Daya

11.6. Kritik terhadap CCPM

Manajemen Proyek Rantai Kritis bukan tanpa kritik. Beberapa argumen yang menentang proses tersebut termasuk tuduhan berikut dan kelemahan yang dirasakan dalam metodologi:

1. Kurangnya pencapaian proyek membuat penjadwalan terkoordinasi, terutama dengan pemasok eksternal, sangat bermasalah. Kritikus berpendapat bahwa kurangnya tonggak

dalam proses proyek mempengaruhi kemampuan untuk mengkoordinasikan tanggal jadwal dengan pemasok yang menyediakan pengiriman eksternal komponen penting.

2. "Kebaruan" CCPM adalah poin yang dibantah oleh beberapa orang yang melihat teknik ini sebagai tidak cocok untuk banyak jenis proyek atau hanya rekonseptualisasi metodologi penjadwalan yang dipahami dengan baik (seperti PERT), asalkan perhatian khusus telah diambil ke tingkat sumber daya jaringan.
3. Meskipun mungkin benar bahwa CCPM membawa peningkatan disiplin untuk penjadwalan proyek, metode yang efisien untuk penerapan teknik ini pada portofolio proyek perusahaan tidak jelas. Metode ini tampaknya menawarkan manfaat berdasarkan proyek per proyek, tetapi kegunaannya di tingkat program belum terbukti. Juga, karena CCPM mendukung sumber daya khusus, dalam lingkungan multi-proyek di mana sumber daya dibagikan, tidak mungkin untuk menghindari *multitasking*, yang mengurangi kekuatan CCPM.
4. Bukti keberhasilan dengan CCPM hampir selalu bersifat anekdot dan berdasarkan studi kasus tunggal. Memperdebatkan manfaat dan perangkap CCPM sebagian besar tetap merupakan latihan intelektual di kalangan akademisi dan penulis teori manajemen proyek. Kecuali untuk pekerjaan pemodelan Budd dan Cooper, tidak ada penelitian empiris skala besar untuk mengkonfirmasi atau tidak mengkonfirmasi kemanjurannya.
5. Tinjauan CCPM berpendapat bahwa meskipun menawarkan beberapa konsep berharga, itu bukan solusi lengkap untuk kebutuhan penjadwalan manajemen proyek saat ini. Para penulis berpendapat bahwa organisasi harus sangat berhati-hati dalam mengecualikan proses penjadwalan manajemen proyek konvensional untuk mengadopsi CCPM sebagai satu-satunya metode untuk perencanaan dan penjadwalan kegiatan.
6. Kritik juga menuduh bahwa evaluasi Goldratt tentang estimasi durasi terlalu negatif dan kritis, menunjukkan bahwa pendapatnya bahwa personel proyek secara rutin menambahkan tingkat besar estimasi durasi aktivitas "*padding*" dilebih-lebihkan.
7. Akhirnya, ada kekhawatiran bahwa Goldratt secara serius meremehkan kesulitan yang terkait dengan pencapaian jenis perubahan budaya di seluruh perusahaan yang diperlukan untuk berhasil menerapkan CCPM. Secara khusus, sementara padding perkiraan aktivitas mungkin bermasalah, tidak jelas apakah anggota tim akan bersedia mengabaikan keselamatan atas permintaan manajer proyek selama mereka melihat kemungkinan sanksi untuk tenggat waktu yang terlewat.

SIMPULAN

Perencanaan proyek Agile menawarkan beberapa keunggulan yang berbeda dibandingkan model tradisional perencanaan waterwall. Untuk proyek dengan serangkaian tujuan tetap dan proses yang dipahami dengan baik, perencanaan waterwall berfungsi dengan baik. Namun, Agile dapat berguna di banyak proyek, dapat mengenali kemungkinan perubahan ruang lingkup dan spesifikasi yang terjadi di tengah siklus pengembangan. Karena merupakan metodologi perencanaan berulang dan inkremental, Agile memungkinkan tim proyek untuk merencanakan dan melaksanakan elemen proyek dalam segmen yang lebih pendek (1-4 minggu), yang disebut Sprint. Akhirnya, Agile menyadari bahwa keberhasilan proyek harus dilihat dari sudut pandang pengguna, sehingga menekankan "suara pelanggan" dan pengembangan fitur produk yang mereka hargai, daripada hanya berfokus pada spesifikasi. Fleksibilitas metodologi dan komitmennya untuk menciptakan nilai bagi pelanggan menjadikan Agile sebagai pendekatan perencanaan proyek alternatif yang baik.

Extreme Programming (XP) adalah teknik pengembangan perangkat lunak yang membawa responsivitas pelanggan Agile ke tingkat yang ekstrem. Dua elemen berbeda di XP adalah penggunaan refactoring, yang merupakan proses berkelanjutan untuk merampingkan desain perangkat lunak dan meningkatkan kode selama pengembangan, daripada menunggu pengujian produk akhir. Fitur kedua dari XP adalah penggunaan pemrograman berpasangan, di mana set programmer bekerja berdampingan untuk mendukung upaya satu sama lain. Pemrograman berpasangan mempromosikan proses kolaboratif dalam membuat perangkat lunak dan membantu mempertahankan penekanan konstan pada kualitas selama pengembangan.

Eli Goldratt mengembangkan Theory of Constraints (TOC) untuk memungkinkan manajer mengidentifikasi dan menyelesaikan kendala sistem. Ada lima langkah dalam metodologi TOC, termasuk: 1) mengidentifikasi kendala sistem, 2) mengeksploitasi kendala sistem, 3) menundukkan segala sesuatu yang lain ke kendala sistem, 4) meningkatkan kendala sistem, dan 5) menentukan apakah kendala baru telah ditemukan dan ulangi prosesnya. Untuk penjadwalan proyek, ide Goldratt berlaku untuk jalur kritis, yang merupakan kendala sistem jaringan. Karena semua penjadwalan dan masalah sumber daya yang terkait dengan proyek

biasanya terjadi karena masalah dengan mempertahankan jalur kritis, penjadwalan Rantai Kritis membantu manajer proyek mengembangkan jadwal yang lebih efektif dan akurat.

Karena masalah sistematis dengan penjadwalan proyek, Eli Goldratt mengembangkan proses Critical Chain Project Management (CCPM). Dengan CCPM, beberapa perubahan dibuat pada proses penjadwalan PERT tradisional. Pertama, semua aktivitas slack individu, atau "buffer", menjadi buffer proyek. Setiap anggota tim, yang bertanggung jawab atas komponen jaringan aktivitasnya, membuat perkiraan durasi yang bebas dari bantalan apa pun, yaitu, yang didasarkan pada kemungkinan keberhasilan 50%. Semua aktivitas pada rantai kritis dan rantai pengumpan (rantai nonkritis dalam jaringan) kemudian dihubungkan dengan bantalan waktu minimal.

Buffer proyek sekarang dikumpulkan dan sebagian dari waktu yang dihemat itu (Goldratt menggunakan aturan praktis 50%) ditambahkan ke proyek. Bahkan menambahkan 50% dari waktu yang dihemat secara signifikan mengurangi keseluruhan jadwal proyek sementara mengharuskan anggota tim untuk tidak terlalu peduli dengan bantalan aktivitas dan lebih banyak lagi dengan penyelesaian tugas.

Manajemen Proyek Rantai Kritis mengasumsikan bahwa rantai kritis untuk sebuah proyek memerlukan identifikasi konflik sumber daya terlebih dahulu dan kemudian urutan tugas untuk menghilangkan konflik ini. Alih-alih menggunakan metode mulai awal untuk jaringan, pendekatan CCPM menekankan penggunaan waktu mulai terlambat, menambahkan buffer feeder di persimpangan jalur feeder ke jalur kritis, dan menerapkan buffer proyek keseluruhan di tingkat proyek untuk digunakan sesuai kebutuhan. Semua aktivitas diurutkan untuk mengeksploitasi konflik sumber daya, memastikan penundaan minimal antara tugas dan mempercepat proyek secara keseluruhan.

DAFTAR PUSTAKA

Jeffrey Pinto (2020). *Project Management Achieving Competitive Advantage*, Fifth Edition, Pearson, ISBN: 978-1-292-26916-0, Chapter 11