Workday Calculations In ACE

ACEIT Users Workshop
National - Public Audience
January 31 – February 2, 2011

John Sandburg - Tecolote Research, Inc.
Antonio Rippe - Tecolote Research, Inc.

Approved for Public Release

Overview

- Motivation
- Calendar vs. Work Days
- Intro to Julian Dates
- Landing on a Work Day
- Effort Durations – Math and Conversion
- Putting it Together Example
- Dealing with Holidays (if time allows)
- In Summary

> Time is the longest distance between two places.
>
> -Tennessee Williams

Motivation

- **Joint analysis of cost and schedule usually housed in a scheduling tool**
  - Detailed schedule with a logically linked target date (e.g. Launch)
  - Usually limited when it comes to defining cost parameters

- **Investigated using ACE as the joint cost & schedule estimating platform**
  - Built-in scheduling functions
  - Robust cost handling
  - Schedule related modeling issues identified

Using built in functions and creating User Defined Functions (UDF) allows ACE to fully replicate and run analysis on a logically linked schedule.

Calendar vs. Work Days

- **ACE works in calendar days**
  - Every day counts—including weekends and holidays
    - Ex. There are 327 days until Christmas
  - Makes math easy for durations and successive days
    - Ex. How many workdays until Christmas?
  - Excel works in calendar days as well
    - Another reason for ACE to do it
- **However, effort is commonly scheduled in workdays**
  - Duration estimates are often in workdays
  - Milestone dates land on weekdays
    - Embarrassing to offer an estimated deadline that lands on Sunday
  - MS Project uses workday calendar
    - Ex. 234 workdays until Christmas (which lands on a Sunday)

Intro to Julian Date

- **We generally think and talk in Gregorian dates**
  - Ex. If we don't finish by 1st of October, we're looking for jobs
- **Computers HATE Gregorian dates because they don't add**
  - Ex. 5/28/2012 + (3/12/2012 – 2/1/2011) = 6/39/2013?

The Julian date (JD) is the interval of time in days and fractions of a day since January 1, 4713 BC Greenwich noon.

- **Computers LOVE Julian dates**
  - They are easy to add and subtract (to work in time spans)
  - They are always valid (e.g., no possibility of 39st September)
- **Many alternatives to official Julian dates**
  - Ex. The Dublin Julian date is days from 31 December 1899
- **ACE uses Excel's definition: Days from 30 December 1899**[1]
  - Makes reporting in POST far easier to do.

[1] Excel mistakenly includes Feb 29, 1900, so 01Jan1900 = 1 and 01Mar1900 = 61.

Landing on a Workday

- **One cool thing about Julian Days is that the day of the week is easy to find**
  - If the Excel Julian Date is evenly divisible by 7, the date is Saturday!

Mod(Julian Date, 7) = 0 → Saturday
Mod(Julian Date, 7) = 1 → Sunday

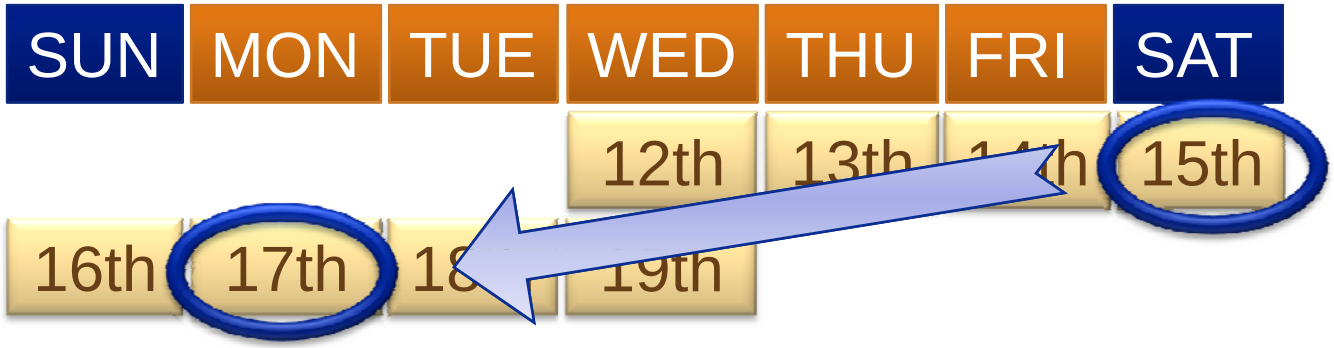
- **This it true all the time, because even on leap years the week is 7 days long (Below shows 'Mod' in action)**

SAT	SUN	MON	TUE	WED	THU	FRI
		863 ↳2	864 ↳3	865 ↳4	866 ↳5	867 ↳6
868 ↳0	869 ↳1	870 ↳2	871 ↳3	872 ↳4	873 ↳5	874 ↳6
875 ↳0	876 ↳1	877 ↳2				

Avoiding Weekends

- **We can write the following ACE UDF to land on workday**
- **Given a date representing a "finish date" for effort, then…**
  - The following UDF takes effort mistakenly assigned to the weekend and pushes it to the workday in the following week.

	WBS/CES Description	Unique ID	Point Estimate	Equation / Throughput
14	*Date Manipulation UDFs			
15	UDF - Pass the Weekend	PW(date)		If(Mod(date,7)<2, date+2, date)
16				
17	*Estimate	*Estimate		
18	Effort landing on Saturday		17MAR2015 *	PW(15Mar2015)



Effort Durations

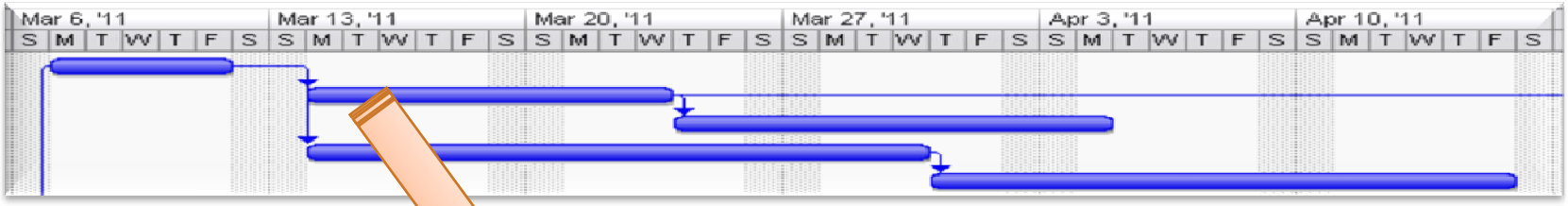
Find a job you like and you add five days to every week.

-H. Jackson Brown, Jr.

- **Durations (time-spans) in ACE are in calendar days**
  - E.g., how many days until Christmas?
- **Be warned: We cannot simply add effort to a date**
  - Durations in work days underestimate when effort is completed
    - E.g., 60 days of effort is 3-months, not 2 months
- **Common adjustment: "CalDuration = WorkDuration * 7 / 5"**
  - This produces rounding error in series of small durations
  - The cumulative effect can become noticeable
- **For schedule work, an accurate calculation is needed**

Finding a Start Date

- **To find a successor's start we use predecessor's finish**
  - To mimic MS Project, the start is one day after the previous finish...



  - Previous task ends Friday and next task begins on Monday

	WBS/CES Description	Unique	Point Estimate	Equation / Throughput	Phasi	Dur (!)	Start Date	Finish Date
14	*Date Manipulation UDFs*							
15	UDF - Pass the Weekend	PW(date)		If(Mod(date,7)<2, date+2, date)				
16	UDF - Successor Start	StD(@pred)		PW(pred.aFinishDate+1)				
19	*Estimate*	*Estimate*						
21	Effort ending on Friday	On_Fri	11MAR2011 *	aFinishDate	C			11Mar2011
22	Workday following Friday		14MAR2011 *	aStartDate	C		StD(@On_Fri)	

  - If we have 'milestone' tasks (0 duration), tweak as follows:

PW(pred.aFinishDate + if(Dur>0,1,0))

Finding a Finish Date

- **A finish date is the start date + duration – 1** *(for non-milestones)*
  - If duration is in calendar days, we're done.
- **What if duration (e.g., effort) is in workdays?**
  - We need to deal with weekends to find the proper finish date.
  - For a duration (d) and start date (st), here is the UDF:

if (d=0, st, PW(st+d-1 + 2*RndDn((d-1+mod(st-2,7))/5)))

	WBS/CES Description	Unique ID	Point Estimate	Equation / Throughput	Phasi	Dur (!)	Start Date	Finish Date
14	*Date Manipulation UDFs							
15	UDF - Pass the Weekend	PW(date)		If(Mod(date,7)<2, date+2, date)				
16	UDF - Successor Start	StD(@pred)		PW(pred.aFinishDate+1)				
17	UDF - Finish after d workdays	FnD(st,d)		if (d=0, st, PW(st+d-1 + 2*RndDn((d-1+mod(st-2,7))/5)))				
19	*Estimate	*Estimate						
20	Adjust landing on Saturday		14MAR2011 *	PW(12Mar2011)	C			
21	Effort ending on Friday	On_Fri	11MAR2011 *	aFinishDate	C			11Mar2011
22	Workday following Friday		14MAR2011 *	aStartDate	C		StD(@On_Fri)	
23	Work 12 days (Mon thru Tue)		29MAR2011 *	aFinishDate	C	12	StD(@On_Fri)	FnD(aStartDate,Dur)

15 calendar days

Breakdown of Finish Date

- **Break it down…**
  - ‘st’ is start date and ‘d’ is effort in work days.
  - Handle special case where duration is 0 (milestone marker).

if (d=0, st, …)

  - Calculate the finish work day without weekends included

if (d=0, st, **PW(st+d-1 …)***)*

  - Each weekend includes 2 days

if (d=0, st, PW(st+d-1 **+ 2 *…***))

  - We include a weekend for every 5 work days

*if (d=0, st, PW(st+d-1 + 2 **RndDn((d-1…)/5)**))*

  - And one extra weekend if starting too far into the first week

*if (d=0, st, PW(st+d-1 + 2 * RndDn((d-1* **+ mod(st-2,7)** *)/5)))*

Date Span to Duration

- **What about calculating workdays between two dates?**
  - Ex. How many workdays in August 2011?
  - UDF: WD(s, f), where ‘s’ is start date and ‘f’ is finish date.

Max(0, (f-s+1) - 2*RndDn((f-s+1)/7) - if(Mod((f-s+1),7)>5-Mod(s-2,7), 2, 0))

Duration in calendar days

Subtract weekends between two dates

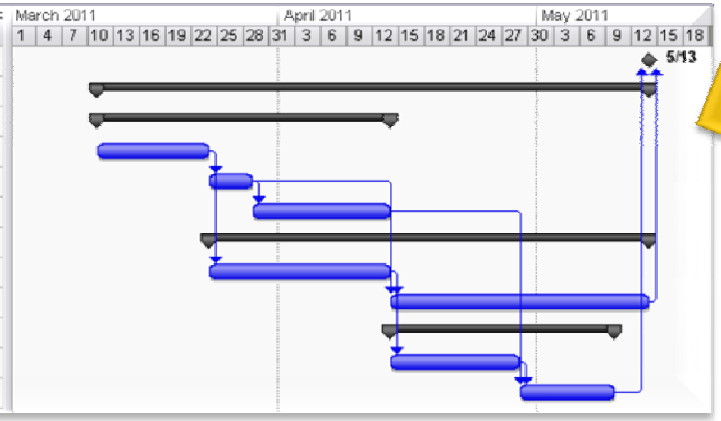
Subtract one extra weekend if necessary

	WBS/CES Description	Unique ID	Point Estimate	Equation / Throughput
18	UDF - Workdays in timespan	WD(s,f)		Max(0, (f-s+1) - 2*RndDn((f-s+1)/7) - if(Mod((f-s+1),7)>5-Mod(s-2,7), 2, 0))
19	*Estimate	*Estimate		
24	No. workdays in Aug 2011		23. *	WD(01Aug2011,31Aug2011)

Chaining Efforts

- **We can now accurately construct Min/Max schedule logic**

Task Name	Duration	Start	Finish	Predec F
Completion	0 days	Fri 5/13/11	Fri 5/13/11	9,12
Project A	**46 days**	**Fri 3/11/11**	**Fri 5/13/11**	
System 1	**24 days**	**Fri 3/11/11**	**Wed 4/13/11**	
Activity X	9 days	Fri 3/11/11	Wed 3/23/11	
Activity Y	3 days	Thu 3/24/11	Mon 3/28/11	4
Activity Z	12 days	Tue 3/29/11	Wed 4/13/11	5
System 2	**37 days**	**Thu 3/24/11**	**Fri 5/13/11**	
Activity X	15 days	Thu 3/24/11	Wed 4/13/11	4
Activity Y	22 days	Thu 4/14/11	Fri 5/13/11	8,5
System 3	**18 days**	**Thu 4/14/11**	**Mon 5/9/11**	
Activity X	11 days	Thu 4/14/11	Thu 4/28/11	8
Activity Z	7 days	Fri 4/29/11	Mon 5/9/11	6,11



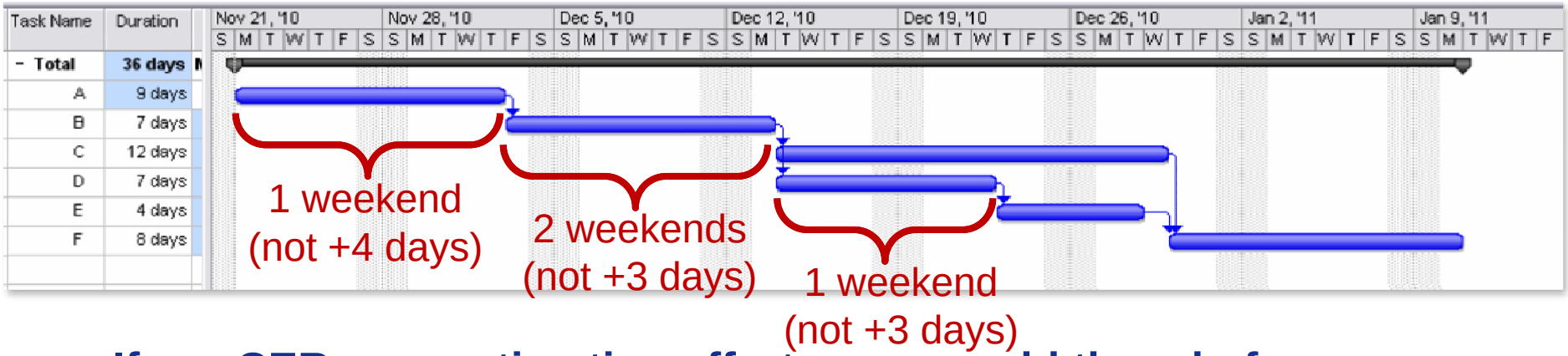
	WBS/CES Description	Unique ID	Point Estimate	Equation / Throughput	Phasi	Dur (I)	Start Date	Finish Date
25								
26	Completion Date	CD	13MAY2011 *	aFinishDate	C	0	Max(StD(@S2_AY),StD(@S3_AZ))	FnD(aStartDate,Dur)
27	Project Duration		46. *	WD(S1_AX.aStartDate, CD.aFinishDate)	C			
28	Project A		*					
29	System 1		*					
30	Activity X	S1_AX	23MAR2011 *	aFinishDate	C	9	11Mar2011	FnD(aStartDate,Dur)
31	Activity Y	S1_AY	28MAR2011 *	aFinishDate	C	3	StD(@S1_AX)	FnD(aStartDate,Dur)
32	Activity Z	S1_AZ	13APR2011 *	aFinishDate	C	12	StD(@S1_AY)	FnD(aStartDate,Dur)
33	System 2		*					
34	Activity X	S2_AX	13APR2011 *	aFinishDate	C	15	StD(@S1_AX)	FnD(aStartDate,Dur)
35	Activity Y	S2_AY	13MAY2011 *	aFinishDate	C	22	Max(StD(@S2_AX),StD(@S1_AY))	FnD(aStartDate,Dur)
36	System 3		*					
37	Activity X	S3_AX	28APR2011 *	aFinishDate	C	11	StD(@S2_AX)	FnD(aStartDate,Dur)
38	Activity Z	S3_AZ	09MAY2011 *	aFinishDate	C	7	Max(StD(@S3_AX),StD(@S1_AZ))	FnD(aStartDate,Dur)

Summary

- The “7 / 5” strategy works well enough for large durations.
- Finding a workday is easy.
- Skipping holidays is hard.
- Serial task effort (durations) takes some extra UDF power.
- Once in place, schedule logic in ACE becomes easier.
- A real-world application of these techniques handles nearly all schedule logic that MS Project provides.
- Using this technique opens up POST and RI$K for schedule-oriented estimates.
- Cost-Phasing in ACE is far more powerful than MS Project

In Addition

Problem with “7 / 5”



- **If our CERs are estimating effort, we can add them before converting to duration in calendar days, just like Project does.**
  - Ex: Gantt chart, critical path is 36 work days and 50 calendar days.
  - Our “7 / 5” trick on total gives us 50.4 ~ 50 calendar days, too.
- **However, if we adjust individual dates and then add, we can run into rounding errors.**
  - Needed when finding maximum of parallel tasks in schedule logic
    - Ex. 13+10+17+11 = 51 calendar days.
  - Reason: Not all efforts span the calculated number of weekend days.
    - Some cover one less weekend than expected,
    - and none cover a half of weekend (e.g., 3 weekend days).

Dealing with Holidays

- **Example of dealing with holidays…**
  - After all of this, we might still land on Christmas in 2013!
  - Most holidays float, making them hard to track
    - Each year lands on different day of month
    - Ex., Labor Day is last Monday in May

WBS/CES Description	Total	FY 2010	FY 2011	FY 2012	FY 2013	FY 2014	FY 2015	FY 2016	FY 2017	FY 2018	FY 2019
Ex. Pass labor day	31MAY2011										
Labor Day for each year	419,698.	31MAY2010	30MAY2011	28MAY2012	27MAY2013	26MAY2014	25MAY2015	30MAY2016	29MAY2017	28MAY2018	27MAY2019

	WBS/CES Description	Uniq	Point Estimate	Equation / Throughput	Ph asi
39					
40	Ex. Pass labor day		31MAY2011 *	PW&H(30May2011)	C
41	Labor Day for each year		*	LD(FYYR)	F
42	Labor Day	LD(year)		MondayOf(DateOf(year,6,1, 1)-7)	
43	Moday on or following date	MondayOf(date)		if(mod(date,7)=2, date, MondayOf(date+1))	
44	Pass weekend & holidays	PW&H(date)		PW(if (date = LD(DateYr(date)), date+1, date))	

Holiday Complications

- **So upon what day does Christmas holiday land?**
  - If you said December 25th, you are wrong.
  - Answer: For the US Government, it lands on the closest workday…
    - possibly the 24th, 25th, or 26th

> "Please note that most Federal employees work on a Monday through Friday schedule. For these employees, when a holiday falls on a non-workday -- Saturday or Sunday -- the holiday usually is observed on Monday (if the holiday falls on Sunday) or Friday (if the holiday falls on Saturday)."
>
> - Office of Personnel Management [1]

	WBS/CES Description	Unique ID	Equation / Throughput	Phasi
	Ex. Christmas holidays		Xmas(FYYR)	F
	Closest Workday	CWDay(date)	If(Mod(date,7)=0, date-1, If(Mod(date,7)=1, date+1, date))	
	Christmas Day	Xmas(year)	CWDay(DateOf(year,12,25,1))	

WBS/CES Description	FY 2010	FY 2011	FY 2012	FY 2013	FY 2014	FY 2015	FY 2016	FY 2017
Ex. Christmas holidays	24DEC2010	26DEC2011	25DEC2012	25DEC2013	25DEC2014	25DEC2015	26DEC2016	25DEC2017

[1] http://www.opm.gov/Operating_Status_Schedules/fedhol/2010.asp